Using Smartphones to Detect Car Accidents and Provide Situational Awareness to Emergency Responders

Chris Thompson, Jules White, Brian Dougherty,
Adam Albright, and Douglas C. Schmidt

Institute for Software Integrated Systems,
Vanderbilt University, Nashville, TN USA
{cm.thompson,jules.white,brian.p.dougherty,
adam.albright,d.schmidt}@vanderbilt.edu

Abstract. Accident detection systems help reduce fatalities stemming from car accidents by decreasing the response time of emergency responders. Smartphones and their onboard sensors (such as GPS receivers and accelerometers) are promising platforms for constructing such systems. This paper provides three contributions to the study of using smartphone-based accident detection systems. First, we describe solutions to key issues associated with detecting traffic accidents, such as preventing false positives by utilizing mobile context information and polling onboard sensors to detect large accelerations. Second, we present the architecture of our prototype smartphone-based accident detection system and empirically analyze its ability to resist false positives as well as its capabilities for accident reconstruction. Third, we discuss how smartphone-based accident detection can reduce overall traffic congestion and increase the preparedness of emergency responders.

1 Introduction

Emerging trends and challenges. Car accidents are a leading cause of death [2]. Automated car accident detection can save lives by decreasing the time required for information to reach emergency responders [6,5,7]. Conventional vehicular sensor systems for accident detection, such as OnStar, notify emergency responders immediately by utilizing built-in cellular radios and detect car accidents with in-vehicle sensors, such as accelerometers and airbag deployment monitors. Figure 1 shows how traditional accident detection systems operate.

Car accident detection and highway congestion control is an emerging application for wireless mobile sensor networks. Recent advances in smartphone technologies are making it possible to detect car accidents in a more portable and cost effective manner than conventional in-vehicle solutions. Rapid accident detection and response can save lives and reduce congestion by alerting motorists as soon as possible, giving them time to reroute. Recent smartphones, such as the HTC Nexus One (an Android-based device), have significantly increased computational abilities compared to previous devices. For example, the Nexus One has a 1Ghz processor and 512MB of RAM compared to the older Palm Treo's 312Mhz processor and 64MB of RAM. The pervasiveness of smartphones also means that the infrastructure required to establish such a wireless mobile

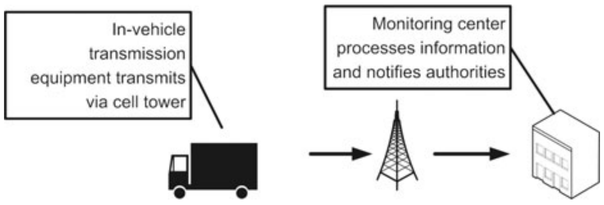

In-vehicle transmission equipment transmits via cell tower
Monitoring center processes information and notifies authorities

Fig. 1. A Traditional Accident Detection System

sensor network is already in place and available after installing appropriate application software.

Smartphone manufacturers also have begun including a plethora of sensors that enable devices to detect the context in which they are being used. For example, the HTC Dream (also an Android-based device), possesses a compass, accelerometer, and GPS receiver allowing application developers to determine the geographic position, heading, and movement of the user. The processing power, popularity, and relatively low cost [12] (compared to other traffic monitoring techniques) make smartphones an appealing platform to construct a wireless mobile sensor network that detects car accidents.

Smartphone-based accident detection applications provide several advantages relative to conventional in-vehicle accident detection systems, *e.g.*, they are vehicle-independent, increasingly pervasive, and provide rich data for accident analysis, including pictures and videos. Building a smartphone-based wireless mobile sensor network for accident detection system is hard, however, because phones can be dropped (and generate false positives) and the phone is not directly connected to the vehicle. In contrast, conventional in-vehicle accident detection systems rarely incur false positives because they rely on sensors, such as accelerometers and airbag sensors, that directly detect damage to the vehicle.

Solution approach → Use onboard sensors and physical context information to detect car accidents. This paper shows how smartphones in a wireless mobile sensor network can capture the streams of data provided by their accelerometers, compasses, and GPS sensors to provide a portable "black box" that detects traffic accidents and records data related to accident events, such as the G-forces (accelerations) experienced by the driver. We also present an architecture for detecting car accidents based on WreckWatch, which is a mobile client/server application we developed to automatically detect car accidents. Figure 2 shows how sensors built into a smartphone detect a major acceleration event indicative of an accident and utilize the built-in 3G data connection to transmit that information to a central server. That server then processes the information and notifies the authorities as well as any emergency contacts.

WreckWatch provides functionality similar to an accident/event data recorder by recording the path, speed, and forces of acceleration on a vehicle leading up to and during an accident [4]. It can also notify emergency responders of accidents, aggregate images and video uploaded by bystanders at the scene of an accident, and send prerecorded text and/or audio messages to emergency contacts. We built WreckWatch using Google Android on the client and Java/MySQL with Jetty and the Spring Framework on the server. The WreckWatch server utilizes custom XML and JSON to communicate with the client applications and the clients use standard HTTP `post` operations to

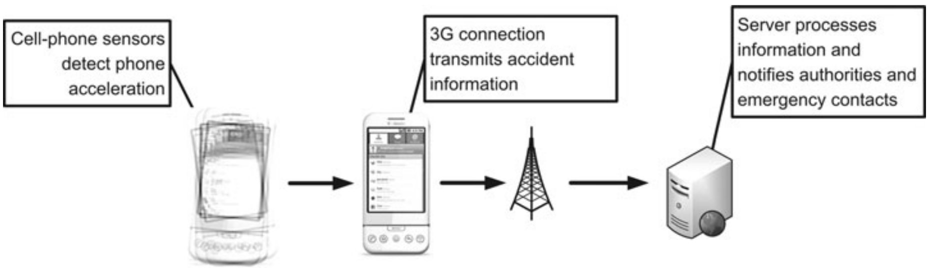

Cell-phone sensors detect phone acceleration
3G connection transmits accident information
Server processes information and notifies authorities and emergency contacts

Fig. 2. Smartphone-Based Accident Detection System

transmit information to the server. WreckWatch also uses a digital PBX running Asterisk to communicate with first responders and emergency contacts.

Paper organization. The remainder of this paper is organized as follows: Section 2 describes the challenges associated with using smartphones to detect traffic accidents; Section 3 describes how WreckWatch overcomes these challenges; Section 4 empirically evaluates WreckWatch's ability to prevent false positives and accident reconstruction capabilities; Section 4 compares our work on smartphone-based accident detection systems with related work; and Section 5 presents concluding remarks.

2 Challenges Associated with Automatically Detecting Car Accidents

This section describes the challenges associated with detecting car accidents via software running on smartphones. A key challenge of developing software to detect collisions is the lack of integration between the smartphone and the vehicle. In contrast, conventional in-vehicle car accident detection systems rely on internal sensors (*e.g.*, airbag deployment sensors) and can assume that any instance of high acceleration/-deceleration is caused by a collision. These assumptions must be rethought by smartphone applications seeking to replace or augment the functionality of conventional in-vehicle systems.

2.1 Challenge 1: Detecting Accident without Electronic Control Unit Interaction

Conventional in-vehicle accident detection systems rely on sensor networks throughout the car and direct interaction with the vehicle's electronic control units (ECUs). These sensors detect acceleration/deceleration, airbag deployment, and vehicular rollover [3,14]. Metrics from these sensors aid in generating a detailed accident profile, such as locating where the vehicle was struck, number of times it was hit, severity of the collision, and airbag deployment.

Smartphone-based accident detection applications must provide similar information. Without direct access to ECUs, however, it is harder to collect information about the vehicle. Although many cars have accident/event data recorders (ADRs/EDRs), it is unrealistic to expect drivers to connect their smartphones to these ADRs/EDRs every time they got in the car, which would require a standardized interface (physical and software) to ensure compatibility. Moreover, while many new cars have some form of ADR/EDR, any smartphone application that required interaction with an onboard computer would be useless in cars that lacked one. It is therefore necessary to collect the same or similar information utilizing only the sensors present on the smartphone device.

Section 3.2 explains how WreckWatch addresses this challenge by using the sensors in the Android platform to detect accelerations/decelerations experienced by car occupants and Section 4 analyzes device sensor data captured by WreckWatch.

2.2 Challenge 2: Preventing False Positives

Vehicle-based accident detection systems monitor a network of sensors to determine if an accident has occurred. Instances of high acceleration/deceleration are due to a large change in velocity over a very short period of time. These speeds are hard to attain if a vehicle is not controlled by a human driver, which simplifies accident detection since we can assume any instance of high acceleration constitutes a collision involving human drivers. Since smartphones are portable, however, it is not as hard to attain such speeds. For instance, it is not hard to drop a phone from six feet in the air, but dropping a vehicle from that height would require significantly more effort.

Since a smartphone-based accident detection application contacts emergency responders—and may dispatch police/rescue teams—it is essential to identify and suppress false positives. Due to smartphone mobility it is hard to programmatically differentiate between an actual car accident versus a dropped purse or a fall on a hard surface. The inability to accurately identify and ignore false positives could render smartphone-based accident detection applications useless by wasting emergency responder resources responding to incident reports that were not car accidents.

Section 3.2 explains how WreckWatch addresses this challenge by using device usage context, such as speed, to filter out potential false positives and Section 4.2 provides empirical results evaluating WreckWatch's ability to suppress false positives.

3 Solution Approach

This section describes the client/server architecture of WreckWatch and outlines the solutions to the challenges presenting in Section 2.

3.1 The WreckWatch Client/Server Architecture

WreckWatch is separated into two main components—the WreckWatch server and the WreckWatch client—that are shown in Figure 3 and described below.

The WreckWatch client acts as a mobile sensor, relays accident information to the server, and provides an interface for third-party observers to contribute information to the accident report. For example, Figure 4 shows how images of an accident can be uploaded to the WreckWatch server. Emergency responders can access the uploaded images via mobile devices en route or a standard web browser at an emergency response center. The WreckWatch client provides mapping functionality through Google Maps on the device to ensure that emergency responders can continuously receive information

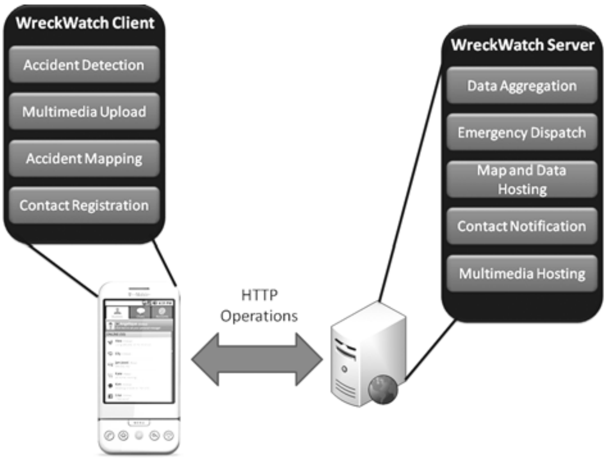


Fig. 3. WreckWatch Architecture Diagram

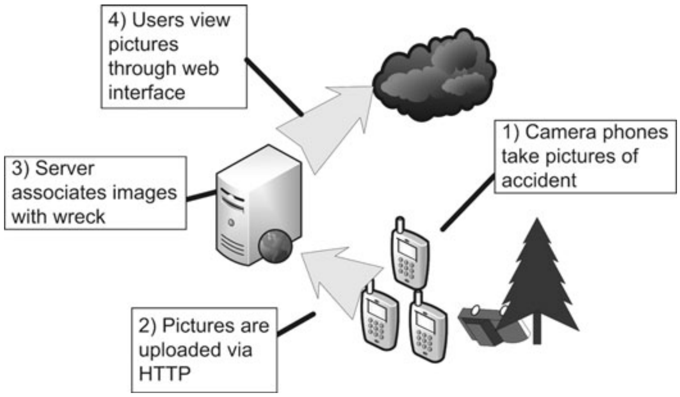

4) Users view pictures through web interface
1) Camera phones take pictures of accident
3) Server associates images with wreck
2) Pictures are uploaded via HTTP

Fig. 4. Accident Image Upload

about an accident to prepare them for whatever they encounter at the accident site. This map also allows other motorists to intelligently route themselves around an accident, thereby reducing congestion.

The WreckWatch Android client is written in Java based on Android 1.5 with Google APIs. It consists of several Android application *activities*[1] for mapping, testing, and image upload. Background services detect accidents by polling smartphone system sensors, such as the GPS receiver and accelerometers. The polling rate is configurable at compile-time to meet user needs and to provide the appropriate power consumption characteristics. The WreckWatch client can gather data from phone databases (such as

[1] Activities are basic building block components for Android applications and can be thought of as a “screen” or “view” that provide a single, focused thing a user can do.

an address book) to designate emergency contacts. Communication to the server from the Android client uses standard HTTP `post` operations.

The WreckWatch server provides data aggregation and a communication conduit to emergency responders, family, and friends. It allows clients to submit accident characteristics (such as acceleration, route, and speed) and presents several interfaces, such as a Google Map and XML/JSON web services, for accessing this information. As accident information becomes available, the WreckWatch server posts location, route and severity information to a Google Map to aid emergency responders, as well as other drivers attempting to navigate the roads near the accident. This map is available over HTTP through a standard web browser and is built with AJAX and HTML, as shown in Figure 5.

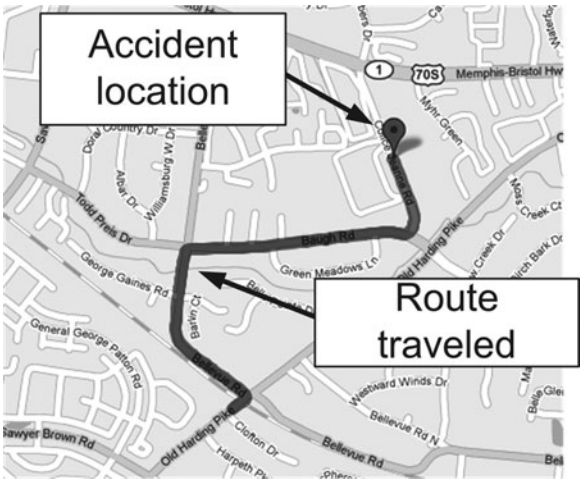


Fig. 5. WreckWatch Accident Map

The WreckWatch server uses digital PBX functionality to make/receive phone calls and provision phone lines dynamically. It can therefore interact with emergency responders via traditional circuit-switched networks and create accident information hotlines in response to serious accidents via an Asterisk-based digital PBX running Linux. The server can also be configured with emergency contacts to notify via text and/or audio messages in the event of an accident. This data is configured at some time prior to a collision event so the server need not interact with the client to notify family or friends.

The WreckWatch server is a web-based service based entirely on freely-available APIs and open-source software. It is written in Java and built using Jetty atop the Spring Framework. It utilizes a MySQL database to store accident information and image meta-information. The server communicates with the clients via a RESTful architecture over HTTP using custom XML (for the Android application) and JSON (for the web-based application).

All communication between the clients and the server is initiated by clients. The server's operations (such as accident information upload) are performed by individual handlers that can be configured at runtime and are specified by parameters in an HTTP request. This architecture enables the addition of new operations and functionality without any software modifications or the need to recompile. All configuration is handled by an XML file that is parsed during server startup.

The PBX is built on Asterisk and connects to the server through a Java API. The Android client and web client pull information from the server and can be configured based on user needs. Due to the loose coupling and use of open standards between clients and server, additional clients for other platforms (such as other smartphones or desktop applications) can be implemented without the need to update the server. The WreckWatch server architecture also supports a heterogeneous group of clients, while providing appropriate qualities of service to each device.

3.2 WreckWatch Solution Implementations

The remainder of this section outlines how WreckWatch addresses the challenges presented in Section 2.

Utilization of Onboard Accelerometers to Detect Collisions. The challenge presented in Section 2.1 explains why it is hard to detect car accidents without ECU interaction. To address that challenge, WreckWatch uses Android's onboard sensors to detect the forces and accelerations associated with a car accident, as shown in Figure 6. The Android platform provides an orientation sensor comprised of three independent accelerometers that allow WreckWatch to detect car accidents in the same manner as vehicle ECUs.

In the event of an accident, the smartphone will experience the same forces and accelerations experienced by the occupants of the vehicle. Moreover, if the smartphone remains stationary relative to the vehicle during the collision, it is possible to use the data gathered from the smartphone to recreate and model the forces it experienced. In this case, the smartphone can provide data much like that gathered by vehicular ECUs.

Smartphones are often carried in some form of pocket [10] attached to a person. In these cases, the smartphone would experience the same forces as vehicle occupants, and

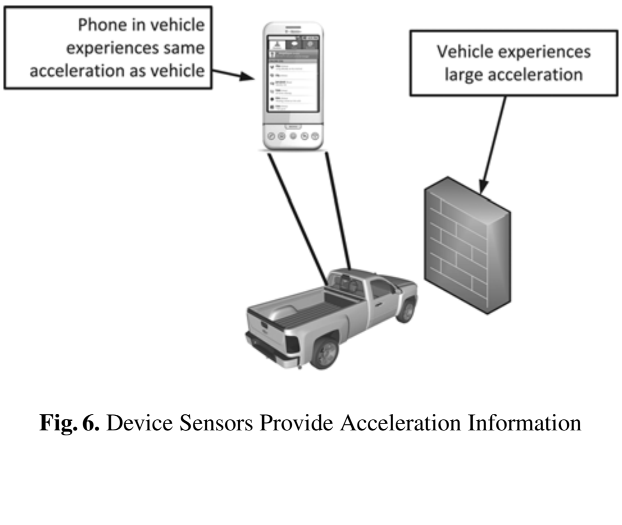

Phone in vehicle experiences same acceleration as vehicle
Vehicle experiences large acceleration

Fig. 6. Device Sensors Provide Acceleration Information

could thus provide more information than in-vehicle systems by recording the forces experienced by occupants rather than just the vehicle itself. When this directionality and movement is combined with speed and location information from the GPS receiver, it is possible to fully reconstruct the accident, including any secondary impacts.

Using Context Information to Eliminate False-Positives. Section 2.2 describes the potential for false positives, which is a key concern with applications that automatically dispatch police or rescue. To address that challenge, WreckWatch employs the following sensor-based and context filters:

– **In order to prevent excessive power consumption and WreckWatch is only enabled when plugged in** GPS receivers consume a substantial amount of power and sampling them at the rate necessary to accurately determine speed would make WreckWatch unusable because it would limit the lifetime of the device to several hours. However, users are able to plug smartphones into cigarette lights in vehicles to provide them with power. Requiring users to plug the smartphone in helps to establish context, which will eliminate false positives, and also mitigates the power consumption of the GPS receiver. However, it is also possible to plug a smartphone in to a wall socket in a home which necessitates additional filters.
– **Speed filter determines whether users are in vehicles.** WreckWatch uses the smartphone's GPS to determine device and (consequently) vehicle speed. However, it only begins recording accelerometer information and looking for potential accidents above 15mph. This filter helps eliminate any acceleration events due to significant accidental smartphone drops that might occur outside a vehicle as well as reducing battery drain. After WreckWatch determines that users are in vehicles, it maintains that as their context until the device is unplugged, which prevents WreckWatch system from shutting off at stop lights. This speed threshold can be adjusted at compile time to prevent overloading operators and falsely alerting family of an accident.
– **Acceleration filter prevents drops and sudden stops from triggering accident notifications.** Filtering alone does not eliminate all false positives, such as a drop inside the vehicle or a sudden stop. To address these issues, therefore, WreckWatch ignores any acceleration events below 4G's. This value is designed to detect even minor accidents but filter out a drop or sudden stop and was chosen based on the empirical analysis presented in Section 4. This threshold is significantly lower than the acceleration required to deploy airbags because of physical environment of the smartphone.

  Accelerometers attached to the vehicle are what trigger airbag deployment. These accelerometers are physically mounted to the chassis of the car meaning that their motion will directly mirror that of the vehicle and will experience every force the vehicle experiences. Smartphones, however, are likely to be held in a pocket or in a cup holder. Car safety systems are designed to reduce the force on the occupants of the car during an accident and because of this, the forces experienced by the phone will be significantly less than the forces experienced by the accelerometers in the car. These systems accomplish this reduction in force by increasing the time over which the change in velocity occurs. The net change in speed is the same, but the acceleration is less because it occurs over a longer period of time. Therefore in order to detect car accidents, the detection threshold must be significantly lower than that required to deploy the airbag. In contrast, the peak accelerations experienced inside of a football helmet during play are approximately 29.2 G's [13]. This value represents the maximum value experienced by a player and would be significantly larger than many minor collisions.

4 Empirical Results

This section describes empirical results of tests performed on the WreckWatch application described in Section 3. These results demonstrate WreckWatch's ability to prevent false positives and gather information to reconstruct an accident accurately.

4.1 Overview of the Experimentation Platform

All experiments were performed on a Google ION device running the vendor image of Android 1.5 on a 525 Mhz processor with 288 MB of RAM. The device was factory reset before loading WreckWatch and no additional third-party applications were installed. WreckWatch recorded acceleration on three axes at the highest possible rate and wrote these values to a CSV file on the SD card in the device. This data was then downloaded to a Windows desktop computer for analysis in Excel.

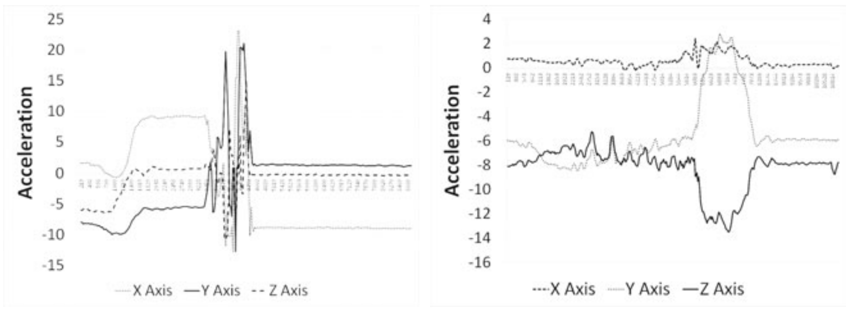
In all graphs, positive z-axis values indicate positive acceleration in the direction from the battery cover toward the screen. Likewise, positive y-axis values indicate positive acceleration in the direction from the USB connector toward the smartphone speaker. Finally, positive x-axis values indicate positive acceleration from left to right when looking at the device with the USB connector closest the observer.

4.2 Evaluating Possibility of False Positives

As described in Section 2.2, avoiding false positives is a key challenge when detecting car accidents with smartphones. To analyze the potential for false positives, we conducted two experiments designed to simulate events that generate accelerations whose values could potentially be interpreted as car accidents. For the first test, the Android device was dropped from ear height in the driver's seat of a car. The device bounced off the seat and wedged between the seat and center console. Figure 7a shows the acceleration on each axis during the collision with the floor.

Using 9.8 m/s as an approximate value for Earth's gravity, the device experienced approximately 2G's in each direction with nearly 3G's on the x-axis before coming to rest. The required acceleration to trigger airbag deployment is 60G's [8,1]. In addition to being $\sim$30 times smaller than required to deploy an airbag, this value is well below the 4G's used as a filter. It is therefore unlikely a smartphone could be dropped in a manner that would exceed 4G's. This data supports the use of a filter as presented in Section 3.2 to prevent false positives.

Another potential scenario that could potentially generate a false positive is a sudden stop. This test was performed in a vehicle by reaching a speed of approximately 25



(a) Acceleration During a Fall (b) Acceleration During a Sudden Stop

Fig. 7. Acceleration During Falls and Sudden Stops

mph and engaging in a sudden stop. The test results are approximate as the exact speed was unknown and braking pressure was not exact. Figure 7b shows the acceleration experienced on each axis during the stop. As described in Section 3.2, because the smartphone remained stationary relative to the vehicle, it experienced the same forces as the vehicle. In this instance, the acceleration experienced by the smartphone was actually less than that experienced during the fall.

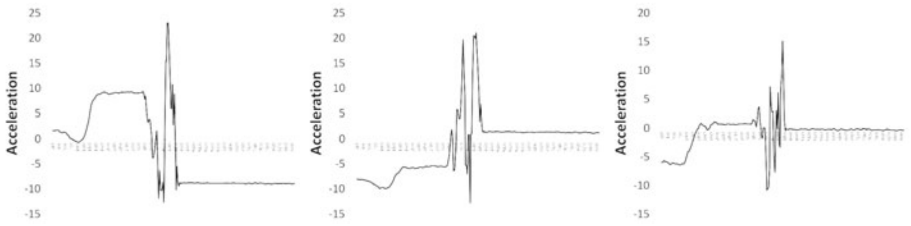
This result is attributed to the fact that although the stop was sudden and forceful, the car (and consequently the smartphone) came to a rest over a period of time that was longer than during the drop test. In other words, the change in velocity was greater but the actual acceleration was less because the change occurred over a longer period of time. Based on this data, it is unlikely for the smartphone to experience 4G's of acceleration simply due to a sudden stop.

4.3 Evaluating Accident Reconstruction Capabilities

WreckWatch can reconstruct an accident based solely on the data gathered from the smartphone. Due to the smartphone's presence in the vehicle during an accident, the smartphone will usually experience the same forces at the same time as the occupants and the vehicle itself. For example, $\sim$40% of cell phones are carried in some form of pocket [10], in which case the device will experience the same forces experienced by the person wearing the pocket.

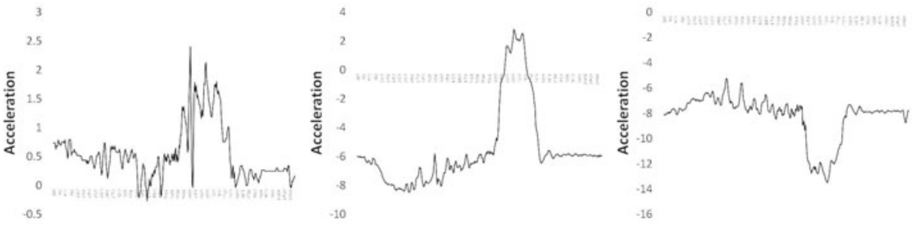
If the smartphone experiences the same forces as the occupants of the vehicle, we can identify what happened during the accident and reconstruct it. To demonstrate this approach, we next analyze the two experiments conducted in Section 4.2.

The graph in Figure 8a shows it is possible to determine that the smartphone was initially experiencing zero acceleration along the x-axis indicating that the x-axis was perpendicular to the ground. This orientation is consistent with holding the smartphone to the ear. While falling, the smartphone tilted such the left edge of the smartphone (relative to the screen with the screen facing away from the ground) was the closest edge to the sky and then flipped again such that the left edge was closest to the ground. When Figures 8a, 8b, and 8c are combined it is clear that the bottom of the smartphone made contact first, followed by the left edge, and finally the back of the device.



(a) X-Axis Acceleration (b) Y-Axis Acceleration (c) Z-Axis Acceleration

Fig. 8. Acceleration During a Sudden Stop



(a) X-Axis Acceleration (b) Y-Axis Acceleration (c) Z-Axis Acceleration

Fig. 9. Acceleration While Dropped in a Car

The acceleration experienced during the sudden stop was actually less than that experienced during the fall. Given what is known about the event, it is therefore possible to identify the orientation of the smartphone during the event. By examining the graphs in Figure 8 it is possible to determine that the smartphone was resting at an angle such that the top of the smartphone was higher than the bottom of the smartphone. The decrease in acceleration along the z-axis is indicative of the force induced on the device by the seat as the car came to a rest. Graphs of other sudden stop events also have a similar appearance so long as the device remained stationary relative to the car.

These reconstruction capabilities give accident investigators the ability to identify what was experienced by the occupants of the vehicle and provide them with information that an ADR/EDR simply cannot provide. This information can also be combined with that present in the ADR/EDR to better understand the entire accident rather than simply the forces experienced by the vehicle itself. WreckWatch gives investigators the capability to analyze a real-world accident in a manner similar to the way they would a controlled collision involving crash-test dummies. Although WreckWatch cannot provide investigators with all impact information (*e.g.*, the forces experienced at the ribs [9] or the pressure on the face [11]), it can provide them with specific information about the overall force on the body and how effectively the restraints protected the passenger.

5 Concluding Remarks

Although conventional in-vehicle accident detection systems provide emergency responders with crucial information at the earliest possible time, adoption of these systems is limited by their non-portability and cost. Smartphones present a promising platform on which to construct an accident detection system. Significant challenges, however, are associated with developing an accident detection application for smartphone-based sensor networks. This paper described how the WreckWatch smartphone application accurately detected traffic accidents by combining (1) contextual information to determine when a user is in a vehicle with (2) high G-force filters that helped to suppress false positives, such as a dropped phone or sudden stop that may occur while a vehicle is in motion.

In developing and evaluating WreckWatch, we learned the following lessons:

• **In the event of an extreme accident the phone may be destroyed preventing it from contacting emergency responders.** As with equipment embedded in the vehicle, which is how systems like OnStar function, there is a chance that the phone would become damaged during an accident and be unable to transmit accident information. Without providing redundant or ruggedized equipment, which would significantly increase cost and reduce usability, there is little that can be done to prevent the destruction of communication equipment. This is a weakness of such a system however the severity of such an accident would likely draw enough attention from witnesses that WreckWatch's notification would be superfluous.

• **Accidents exert extreme forces on a phone that are unlikely to occur when dropping it.** The forces experienced during a car collision are extreme and highly unlikely to occur in any other event other than a high-speed collision. These events are therefore easier to identify and categorize accordingly. Moreover, by combining the accident detection process with contextual information to determine when the user is in a vehicle, false positives are less likely.

• **Smartphones can surpass the functionality of conventional in-vehicle accident detection systems.** Modern smartphone platforms possessing a GPS receiver and accelerometers can be utilized to detect car accidents and represent a portable alternative to conventional in-vehicle systems, such OnStar. Moreover, smartphone-based applications can surpass the functionality of conventional systems by leveraging the other device features and network functionality, such as contact management and Internet access, which allows accident victims to alert emergency personnel, family, and friends immediately and automatically.

• **Collision events can be modeled based on data collected from a smartphone.** If the smartphone remains stationary relative to the vehicle during the collision, the smartphone will experience the same forces as the vehicle, which allows reconstruction of the accident based on the data gathered from the smartphone. This data allows accident investigators to determine not only what happened during an accident, but also provides them with insight into the forces experienced by the occupants. In this case, a smartphone-based accident detection system provides more information than a system like OnStar that only collects information about the vehicle itself. This data could then be used to analyze the effectiveness of the safety features of the vehicle, such as seat belts.

• **It may not be possible to detect all accidents with smartphones.** Due to the filters utilized to prevent false positives, it may be possible to experience a low speed "fender-bender" without the application detecting it. More work is needed to enhance the filtering mechanisms to account for these types of collisions. In particular, WreckWatch's filtering algorithm could be enhanced to determine whether the user is in a vehicle or not utilizing history information. For example, users often travel similar routes to work and WreckWatch could learn where stops or reductions in speed are common by analysis of trends (*e.g.* if a person usually travels through an area at 40mph but occasionally slows to a stop indicating a potential traffic jam). Likewise, WreckWatch could use known intersections to identify potential stops and anticipate them or download traffic information to predict the location of traffic jams resulting from long-duration reductions in speed.

• **In-vehicle Bluetooth radios connecting the phone and vehicle increase the potential for smartphone-based accident detection systems.** Although WreckWatch does not rely on any interaction with the vehicle, direct interaction with the ADR/EDR would increase the accuracy and information available to smartphone-based accident detection systems, such as whether brakes were applied and at what pressure, whether the occupants were wearing seat belts, whether cruise control was on, whether head lamps were on, etc [4]. Many vehicles already possess a hardware connection to the ECU for problem diagnosis. This connection could be used to attach a Bluetooth transmitter that would establish a wireless connection to WreckWatch when the vehicle was started. Minor modifications to WreckWatch would be needed to record and process the additional sensor information from the vehicle.

• **Integrating accident detection systems with Intelligent Transportation Systems (ITS) can help city planners and motorists combine accident data with other roadway information.** City planners and transportation departments currently use ITS to identify road problems and hazardous conditions. Many cities offer services (such as a 511 telephone number) to allow motorists to access information regarding congestion and accidents on major roadways. Integrating WreckWatch with ITS implementations would reduce the latency between an accident event and the availability of the information. This integration could also help city planners create a database of accident locations that could be cross-referenced with hazardous road conditions. Since WreckWatch uses open standards an application that already performs many ITS services could be configured to download accident information using XML over HTTP. This information could then be incorporated into reports generated by the ITS and processed accordingly.

The WreckWatch application is open-source and can be downloaded from `vuphone.googlecode.com`. Also available from this repository are smartphone applications for social networking, campus dining, and social events.

References

1. National Highway Traffic Safety Administration. Federal Motor Vehicle Safety Standards: Occupant Crash Protection - Supplemental Notice of Proposed Rulemaking (1999)
2. National Highway Transportation Safety Administration. 2007 Traffic Safety Annual Assessment - Highlights (2008)
3. Alsliety, M.: How does SDR fit the telematics model?
4. Askland, A.: Double Edged Sword That Is the Event Data Recorder. The Temp. J. Sci. Tech. & Envtl. L. 25(1) (2006)

5. Champion, H.R., Augenstein, J., Blatt, A.J., Cushing, B., Digges, K., Siegel, J.H., Flanigan, M.C.: Automatic crash notification and the URGENCY algorithm: Its history, value, and use. Advanced Emergency Nursing Journal 26(2), 143 (2004)
6. Drabek, T.E.: Managing the emergency response. Public Administration Review 45, 85–92 (1985)
7. Evanco, W.: The Impact of Rapid Incident Detection on Freeway Accident Fatalities. Mitretek Systems, Inc. WN96W0000071 (1996)
8. Fildes, B., Newstead, S., Barnes, J.S., Morris, A.P.: Airbag effectiveness in real world crashes (2001)
9. Groesch, L., Netzer, G., Kassing, L.: Dummy for car crash testing, US Patent 4701132 (October 20, 1987)
10. Ichikawa, F., Chipchase, J., Grignani, R.: Where's the phone? a study of mobile phone location in public spaces. In: Proc. IEE Mobility Conference 2005, Citeseer (2005)
11. Mellander, H., Nilsson, S., Warner, C.Y., Wille, M.G., Koch, M.: Load-sensing faceform for crash dummy instrumentation, US Patent 4691556 (September 8, 1987)
12. Mohan, P., Padmanabhan, V.N., Ramjee, R.: Nericell: Rich monitoring of road and traffic conditions using mobile smartphones. In: Proceedings of the 6th ACM conference on Embedded network sensor systems, pp. 323–336. ACM, New York (2008)
13. Naunheim, R.S., Standeven, J., Richter, C., Lewis, L.M.: Comparison of impact data in hockey, football, and soccer. The Journal of Trauma 48(5), 938 (2000)
14. Verma, M., Lange, R., McGarry, D.: A Study Of US Crash Statistics From Automated Crash Notification Data (2007)