Towards City-Scale Smartphone Sensing of Potentially Unsafe Pedestrian Movements

Trisha Datta
Hillsborough Township Public School
Hillsborough, NJ, USA
tdatta@htps.us

Shubham Jain
WINLAB, Rutgers University
North Brunswick, NJ, USA
shubhamj@winlab.rutgers.edu

Marco Gruteser
WINLAB, Rutgers University
North Brunswick, NJ, USA
gruteser@winlab.rutgers.edu

Abstract—**This paper proposes large scale collection of pedestrian movement data to promote pedestrian safety in our rapidly developing urban environments. As a first step, we develop and test algorithms for sensing unsafe pedestrian movements. With distracted pedestrian fatalities on the rise, and larger than ever use of smart devices, we propose to use the smartphone to protect pedestrians by leveraging the in-built inertial sensors on the smartphone. We discuss how to use these sensors for recognizing user movements that could be potentially risky when walking on the street, while also accounting for different phone orientations. We introduce a simple path prediction technique and use this to compute potential street crossings. In order to evaluate our algorithms, we conducted walking trials and collected data from all relevant sensors. Initial tests indicate a 90.5% success rate in predicting that a pedestrians trajectory will cross a road.**

I. Introduction

As traffic risks, especially those to pedestrians [8], continue to increase, an urgent need arises to understand the causes that lead to pedestrian accidents. In 2011, the number of pedestrian fatalities rose to 4,432, and 69,000 pedestrians were injured [4] [1]. Many of these injuries and deaths are the result of distractions in the form of smartphones [3]. Around 85% percent of people said they had witnessed someone using a phone while walking, and of this 85%, 42% said the pedestrians using a phone bumped into something, and 34% said the pedestrians using a phone put themselves in the path of a moving vehicle [3]. More startling, however, is the fact that more than half of all pedestrians know that texting while walking is dangerous, but more than a quarter of pedestrians still do so [10]. Given that they are unlikely to remedy their behavior, there is an obvious need to develop technologies to address this issue and enhance pedestrian safety.

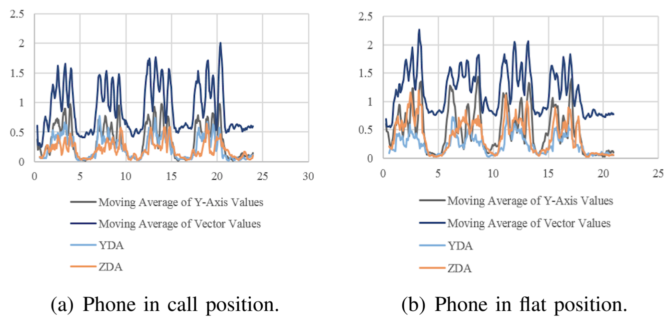
One way to alleviate this problem is to collect large scale pedestrian walking data, identify their behaviors and analyze them. We already have the largest deployment of sensors throughout every city - the smartphones. We suggest tapping into the sensing power of the smartphone at a much larger scale. The smartphones carried by pedestrians in their pockets, are capable of sensing and collecting individual motion. This personal sensing, when aggregated, can provide us with a fresh perspective on pedestrian safety. Our goal is city wide collection of such data and deploying this knowledge to create safer streets for pedestrians.

We hope to leverage the sensors on the phone to keep track of pedestrian movements. Since most pedestrian accidents occur at intersections or midblock locations [6], our aim is to identify events and movements when a person might be entering the street at intersection or at midblock location, which usually happens when people are trying to cross the street. The primary challenge is to determine when people are about to cross the street, or when they turn towards a street to cross it. We isolated a few cases that indicate dangerous movements: Turning toward a road, preparing to cross a road, and passing near an intersection. To detect these events, we need to use phone sensors to detect when a user is turning, moving/stopping, what side of the road a user is on, and the pedestrian's path. We develop algorithms to sense these movements and classify them as safe or unsafe. We used the gyroscope and compass direction to detect turns, linear acceleration to detect movement, GPS location to detect side of the road, and GPS location and compass to predict a pedestrian's path. In addition, once we identified the relevant sensors and behaviors, we studied how accurately we can predict a pedestrians path and its intersection with any roads.

II. Related Work

Wang et. al. [13] have developed a pedestrian safety smartphone app called WalkSafe, which is an application that uses a phones camera to detect if cars are coming toward the user. Though this can be extremely valuable, it is only helpful with the phone held in certain positions, such as when the user is on a call. If a person is texting, which is a far more dangerous activity from the perspective of requiring greater engagement from the user, the camera would probably be pointing at the ground and have no way of detecting approaching cars. Several studies have been done that use anisotropic magneto-resistive (AMR) magnetometers mounted in the road to monitor traffic [12]. These studies have shown it is possible to use a magnetometer, which phones do have, to detect not only that a car has approached but also what direction the car is approaching from and what kind of car is approaching. However, magnetometers in phones are not nearly as sensitive as AMR magnetometers, and our preliminary experiments show that phone magnetometer readings only change significantly when the car is a few inches away, which would be far too late to alert the pedestrian of the danger.

Gandhi et. al. [9] suggested the use of video, radar and laser distance measurement approaches for pedestrian safety. Fackelmeier et. al. [7] discussed an RFID based approach. However, these approaches need significant infrastructure and they work only on individual level, without providing the large scale behaviour tracking. Jain et. al. [11] have shown



(a) Phone in call position. (b) Phone in flat position.

Fig. 1. Movement detection with phone in different positions.

that the GPS on smartphones can be used for pedestrian risk detection in suburban environments. We enhance this approach by exploring the role of inertial sensors for unsafe movement detection.

III. TECHNICAL APPROACH

Our focus is on using the inertial sensors on the smartphone to determine sudden unanticipated movements, which could lead a pedestrian into a potentially dangerous situation. A few such movements include walking outdoor for extended periods, turning towards a street, approaching an intersection. We approach this problem by targeting the sensors on the smartphone and obtaining useful pedestrian context from each of them. An additional concern would be the position or orientation of the phone. We consider common phone positions in our analysis. One of them, the *call* position, is the one where the phone is in a position as if the user is on a call. The seconds, the *flat* position, is the one where the phone is held in a texting position.

We start by using the GPS on the smartphone to determine a pedestrians exact location. Next, we evaluate the pedestrians state of motion (standing, walking) using the accelerometer. Additionally, we use the gyroscope to detect when users change their walking path to make a turn. If the turn is towards a street, we mark it as an unsafe turn, since it will lead them to a moving vehicles path. In the following subsections, we describe each of these sensors and discuss their relevance.

A. Detecting user movement

Detecting user movement is important to understand when a user is walking for an extended period of time and is probably on a road or sidewalk. Similarly, if we determine that a user was moving and has now stopped, then based on their current location, we can determine that he/she is not in potential danger anymore. Therefore, if we can tell that a user who has been walking for a while has stopped near an intersection, we can predict that they are about to cross the road. In order to detect if a person is walking or not, we use their linear acceleration. Linear acceleration is computed by taking the accelerometers readings (the force of acceleration along the phones x, y, and z axes) and filtering them to exclude the force of gravity. To find out when a person is moving, we use linear acceleration readings and simple threshold technique. We walked our testbed with the phone in *flat* and *call* positions and recorded linear acceleration data. There are four values derived from linear acceleration values that we decided to use

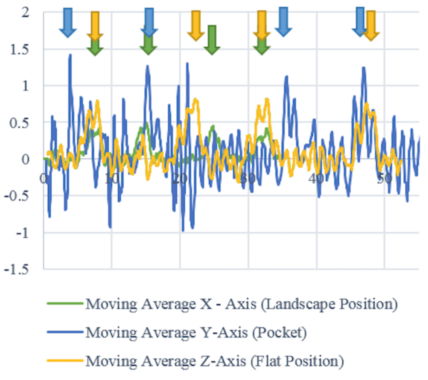


Fig. 2. Left turn detections using gyroscope.

as threshold to determine user movement. The first is a moving average of the last five y-axis readings; the threshold for this was found to be 0.21 m/s^2, as in Figure 1. The next value we used was a moving average of the last five vector values (the vector is the square root of the sum of the squares of the x, y, and z values); the threshold for this was found to be 0.75 m/s^{2}2.

The third value we used was a moving average of the absolute value of differences between consecutive y-axis readings (YDA); the threshold for this was determined to be 0.16 m/s^2. The last value we used was a moving average of the absolute value of differences between consecutive z-axis readings (ZDA); the threshold we computed was 0.15 m/s^2.

B. Turn detection

Turn detection is an important movement. If a pedestrian turns towards a street, he could potentially walk off into a moving vehicle's path. Turn detection can be approached in two ways,using two separate sensors. We explore both, the gyroscope and the compass for detecting when a pedestrian turns. We suggest combining the two, which provides a more robust turn detection.

1) Using gyroscope: A gyroscope can detect turns very quickly, usually whilst the person is in the middle of turn. Looking at the moving average of the x-axis readings for right turns, in Figure 3 and left turns 2 taken while the phone was in the 90° landscape position (see Figure 4 for an explanation of what the 90° landscape position is), we deduced that the threshold for a left turn in this position was 0.4 radians/second and that the threshold for a right turn in this position was -0.4 radians/second. Using this, we deduced that the thresholds for the phone in 270° landscape position (see Figure 4 for an explanation of what the 270° landscape position is) were the opposite values: -0.4 radians/second for left turns and 0.4 radians/second for right turns.

Looking at the moving average of the y-axis readings for right and left turns taken while the phone was in a pocket in Figure 3 and Figure 2, we deduced that the threshold for a left turn in the pocket position was 0.8 radians/second and that the threshold for a right turn in the flat position was -0.8 radians/second. Looking at the moving average of the z-axis readings for right and left turns taken while the phone was flat, we deduced that the threshold for a left turn in the flat

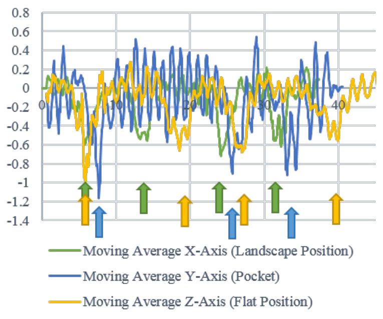


Fig. 3. Right turn detections using gyroscope.

position was 0.5 radians/second and that the threshold for a right turn in the flat position was -0.5 radians/second.

2) Compass for turn verification: he gyroscope is very sensitive to small movements. In order to confirm readings from the gyroscope, we used compass (fusion of accelerometer and magnetometer) readings. Since the compass gives an absolute bearing, we can use it to tell which way a user is facing and by exactly how much he/she may have turned. We use the accelerometer and magnetometer readings as input and compute the rotation around the phones z-axis (azimuth) with respect to north, which essentially corresponds to compass heading. For turn detection using the compass, we record the initial bearing. All subsequent bearings are subtracted from that initial bearing, and if the difference is between 85° and 90° , a turn is confirmed by the compass, and the initial bearing is reset. The axes of the phone do not switch as the phone moves. However, when the phone is held in different positions, we need to remap the phones coordinate system to the real world coordinate system. We remap the phones coordinate system by specifying to which real world axes to map the phone x- and y-axes. For example, when the phone is held vertically, the phone x-axis still maps to the real world x-axis. However, the phone y-axis now maps to the real world z-axis, and the phone z-axis maps to the real world y-axis. Similarly, if the phone is in the 90° orientation position, the phone x-axis is mapped to the real world z-axis and the phone y-axis is mapped to x-axis in the negative direction. The last general phone orientation, the phone in the 270° orientation position, is accounted for by mapping the phone x-axis to the real world z-axis in the negative direction and the phone y-axis to the x-axis.

3) Combining Gyroscope and Compass: We use both the gyroscope and the compass for turn verification. The turn detection algorithm gets a confirmation from the gyroscope, and for three seconds, it waits for a turn confirmation from the compass. If and only if confirmation is received from both turn detection methods, a turn is recorded.

C. Gravity sensor for screen orientation

When using the turn detection algorithm, it became clear that changing the screen orientation sometimes resulted in recording a false turn. To remedy this, we looked at gravity sensor readings. First, we consider the false left turn recorded when a phone switches from the 0° to 90° position. If the phone is held in the 0° position vertically, gravity will be acting positively on the y-axis. If the phone switches to the

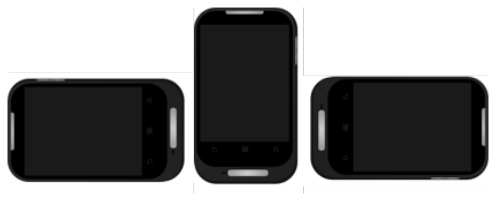
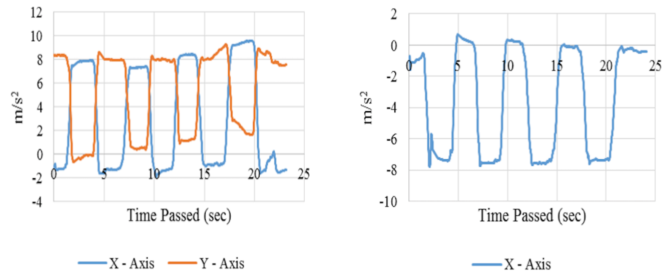


Fig. 4. 90° position, 0° position and 270° position.



(a) Gravity sensor turning from 0 to 90 degrees.

(b) Gravity sensor turning from 0 to 270 degrees.

Fig. 5. Gravity sensor for screen orientation.

90° position, gravity will be acting positively on the x-axis. Figure 5(a) shows x- and y-axis readings from the gravity sensor while the phone is switched from the 0° to 90° position and back to the 0° position four times. Usually, turning from 0° to 90° would result in our turn detection algorithm detecting a left turn, which is not the desired outcome because only the phone changed direction and not the user. Looking at the graph, it is clear that every time the orientation switches, the x- and y-axis values switch places. Thus, we modified the algorithm to only verify a left turn in the 0° position if the y-axis value is larger than the x-axis value at the time of the detected turn. This means that the phone is still in the 0° position and not changing orientation. Similarly, if a left turn is recorded in the 90° position, the application checks to see if the y-axis readings are larger than the x-axis readings, meaning that a screen orientation change is occurring, and if the y-axis readings are larger, no turn is recorded We also had to consider turning from the 0° to 270° position. Figure 5(b) shows the readings from the gravity sensor when the phone was turned from the 0° to 270° position and then back to 0° four times. This would usually result in our turn detection algorithm detecting a right turn. When the phone goes from 0° to 270°, the x-axis value always dips below -5 m/s^2. Thus, when a right turn is recorded in the 0° position, the application waits 0.3 seconds to see if the x-axis value dips below -5 m/s^2, and if it does, no turn is recorded. Similarly, if a left turn is confirmed in the 270° position, then the application checks to see if the x-axis reading goes above -5 m/s^2, and if it does, no turn is recorded. Android does have a method to detect if the phone is in the 0°, 90°, or 270° position, but the advantage of using gravity readings rather than this method is the fact that it detects orientation changes before the built-in method, thus decreasing response time.

D. Path prediction and road intersection

Crossing streets is particularly dangerous. Hence, we are also concerned with advance prediction of when a person

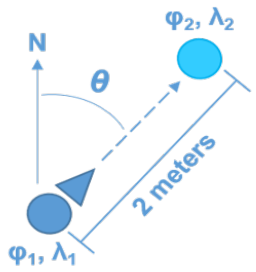


Fig. 6. Path prediction technique.

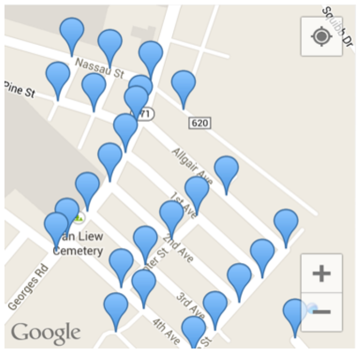


Fig. 7. Intersection computations using OpenStreetMap.

might cross the street. To determine if a person is about to cross the road, we need to predict their future path. We do this using the compass bearing, derived from magnetometer and accelerometer data, and seeing if they have turned left or right. We use the Haversine formula that uses the bearing, a set distance, and current latitude/longitude coordinates to predict future coordinates.

Figure 6 provides an illustration of the path prediction technique, where ϕ is latitude, λ is longitude, θ is the bearing clockwise from north and d is the distance away from the current position, we can predict the latitude/longitude coordinates of the user after he/she has moved two meters. We use two meters because it is long enough to predict a potential crossing with sufficient notice, but it is short enough to not return false positives. We use OpenStreetMap data (described in more detail later) to determine the latitude/longitude coordinates of the road, and we use a line segment intersection algorithm to determine if the predicted path and the road intersect at any point. If such an intersection occurs, we predict a crossing.

E. Intersection location

We also want to identify the cases when a pedestrian is approaching an intersection. In order to locate upcoming street intersections, we use data from OpenStreetMap, which in an open sourced maps database [2]. OpenStreetMap makes it possible for users to export data within a certain geographical region in an XML file. For our purposes, the XML file contains nodes, which are just latitude-longitude points, and ways, which are made up of nodes. Ways correspond to some sort of path, like a footpath or road. To use the OpenStreetMap XML file to find intersections, we first identify all nodes in the selected area and compile a list of their identification numbers. After confirming that a way is a road (by using tags in the

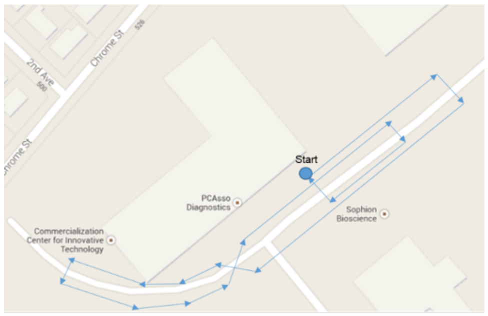


Fig. 8. Test path for walking.

document), we identify the nodes that make up that way. For each node in a way, we add the identification number of the way to a list that maps to the list of identification numbers of the individual nodes. If at the end of the parsing one node is found to be part of more than one way, then we have found an intersection. Figure 7 shows intersections found from OpenStreetMap data in a small sample region.

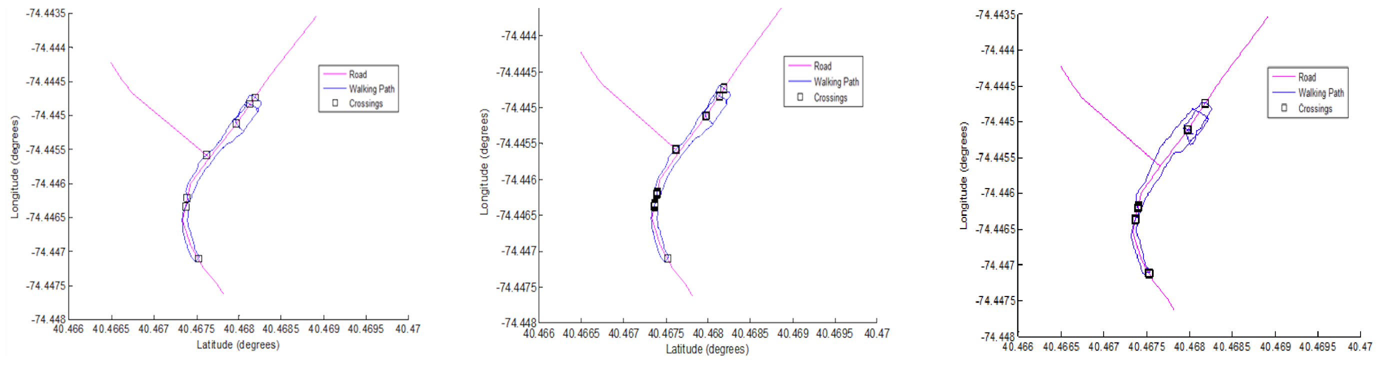
IV. Evaluation

A. Data Collection

To test our algorithms ability to predict user crossings and detect sudden unsafe movements, we walked the path indicated in Figure 8 while holding the phone in the flat position. In order to analyze the data we walked on a test path in New Brunswick, New Jersey. This was a small residential area around our research lab. While doing so, we collected data from the phones accelerometer, magnetometer, gyroscope, gravity sensor, linear acceleration sensor, and GPS. This data was then analyzed offline in MATLAB.

B. Data Analysis

We used Androids methods to calculate the compass bearing with every accelerometer/magnetometer update and saved the bearing to a file coupled with the most recent GPS location update. Next, we used OpenStreetMap data to graph the roads in the walking path area in MATLAB. Afterwards, we used the line segment intersection algorithm on the walking path and the road. Lastly, we used the line segment intersection algorithm on predicted paths and the road. We predicted a new user path every time there was a new available set of magnetometer and accelerometer data; we predicted this path using the new compass bearing and the most recent GPS location update. Because the GPS updates slower than the accelerometer and magnetometer, we did have several compass bearings coupled with one GPS location update. We then parsed an OpenStreetMap XML file that contained the area in which our walking route was located and graphed the relevant roads. When graphing the roads (indicated by ways), we drew line segments from node to node on the graph; these line segments are used later in our algorithm. We first used the line segment intersection algorithm on the walking path latitude/longitude coordinates themselves. We considered every GPS location update and the subsequent update; these two GPS points served as the endpoints of our line segment. We then used the intersection algorithm to see if the walking path intersected with any roads provided by OpenStreetMap.



(a) Detected crossing for walking path trial one. (b) Predicted crossing for walking path trial one. (c) Predicted crossing for walking path trial two.

Fig. 9. Crossing detection evaluation for walking path trial.

Figure 9(a) shows the intersections found by the segment intersection algorithm in one of our trials; black squares mark the points of intersections. We then used the line segment intersection algorithm a second time, this time in conjunction with our path prediction algorithm. The endpoints of our line segment were now every GPS location update and the latitude/longitude point returned by the path prediction method. Since several compass bearings were coupled with each GPS location update, we did have several predicted paths originating from one GPS update. If any intersections were predicted, we graphed a black square at the point of intersection. Figure 9(b) and Figure 9(c) show the predicted intersections found.

All actual crossings from the walking path GPS locations were detected by the line segment intersection algorithm, which is a 100% crossing detection rate, and using the path prediction algorithm on three trials, 19/21 intersections were predicted, which indicates a 90.5% accuracy rate. Some of the inaccuracies were due to the slow rate of GPS update on the smartphone.

V. Discussion and Future Work

Preliminary tests indicate success in detecting movements useful for tracking pedestrian safety, such as when they are turning left or right and predicting when they are about to cross a street. We have demonstrated that data obtained from multiple sensors on the smartphone is useful in predicting pedestrian movements that could be potentially unsafe. We plan to use our turning and stopping/starting sensing algorithms to detect pedestrian behavior that indicates crossings even before our path extrapolation does. Also, we plan to implement a real-time app to perform all of the above. Our tests were conducted in a residential area with not many high-rise buildings. Our results might be affected in larger cities, where GPS accuracies get worse.

When collected from a large number of pedestrians onto a cloud server, these statistics can help identify unsafe locations, such as intersections or road segments in a city. Maps can be created that mark these zones, where pedestrians often indulge in perilous movements, and hence are at higher risk. This data can help authorities create crosswalks and impose other traffic regulations in specific areas. This approach promotes safe infrastructure for pedestrian safety. Moreover, such knowledge discovery also opens space for pedestrian navigation applications. Pedestrians can be navigated through less crowded, safer zones. If cars are also connected to the cloud, real-time warnings could be issued to drivers about potentially unsafe pedestrians in their path. This can be achieved via a technology such as the DSRC [5]. This approach can also be developed further to support safety for bicyclists. This can be particularly important in areas with no dedicated bike lanes. On a personal scale, this combination of location and walking behaviours can help parents monitor their child's walking habit, and whether or not they cross at the designated crossing, or if they dashed across the street.

Smartphone sensing is a powerful tool we carry in our pockets everyday, all the time. We believe that it can be effectively used for building safer environments for pedestrians.

References

[1] Injury prevention and control motor vehicle safety. http://www.cdc.gov/motorvehiclesafety/pedestrian_safety/factsheet.html.

[2] Openstreetmap. http://www.openstreetmap.org/.

[3] Phones put pedestrians in a fog. http://www.consumerreports.org/cro/magazine/2012/08/phones-put-pedestrians-in-a-fog/index.htm.

[4] Traffic safety facts. http://www-nrd.nhtsa.dot.gov/Pubs/811767.pdf.

[5] U.S. Department of Transportation, DSRC. http://www.its.dot.gov/factsheets/dsrc_factsheet.htm.

[6] Federal Highway Administration, Crash-Type manual for pedestrians, April 1997.

[7] Andreas Fackelmeier, Christian Morhart, and Erwin Biebl. Dual frequency methods for identifying hidden targets in road traffic. In *Advanced Microsystems for Automotive Applications*. 2008.

[8] Transportation for America. Dangerous by design, 2011.

[9] T. Gandhi and M.M. Trivedi. Pedestrian protection systems: Issues, survey, and challenges. *Intelligent Transportation Systems, IEEE Transactions on*, 2007.

[10] Liberty Mutual Insurance. Study shows three out of five pedestrians prioritize smartphones over safety when crossing street. http://goo.gl/XuHtX1, June 2013.

[11] Shubham Jain, Carlo Borgiattino, Yanzhi Ren, Marco Gruteser, and Yingying Chen. On the limits of positioning-based pedestrian risk awareness. MARS, 2014.

[12] Lucky S. Withanawasam Michael J. Caruso. Vehicle detection and compass applications using amr magnetic sensors. Sensor Expo Proceedings 1999.

[13] Tianyu Wang, Giuseppe Cardone, Antonio Corradi, Lorenzo Torresani, and Andrew T. Campbell. Walksafe: a pedestrian safety app for mobile phone users who walk and talk while crossing roads. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*, HotMobile '12.