Review of smartphone based applications used to monitor traffic

Literature review

William Lumala
University of Cape Town
lmlwil001@myuct.com

ABSTRACT

Previous research has shown that traffic accidents are one of the leading causes of fatalities in both developed and developing regions. The fatalities in some cases are increased by the time between the accident and when emergency medical personnel arrive at the scene. In a bid to reduce on this time, some developers have developed smartphone applications that can detect car accidents and notify the first responders. Smartphone applications are considered a formidable solution. This is because of ubiquitous usage of smartphones coupled with the fact that services that provide safety and emergency help such as OnStar are not affordable to everyone. This paper analyses and evaluates accident detection applications that have been developed with special attention given to the sensors that are used in the accident detection. In addition to this, the paper points out some areas in this field where improvements can be made based on the evaluation done. The analysis shows that the leading challenge for these applications is the prevention of false positives i.e. the smartphone detecting an accident when it has not occurred. This is an even bigger problem if the application provides notification services. This paper concludes with how the novel idea of using the camera on smartphones can be used to improve on the current work that has been done in this field.

Keywords

Android development, Accident Detection, Smartphone Applications, Sensors, Triggered sensing

© 2016 ACM. ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

1. INTRODUCTION

This paper is about how Android based smartphone applications can be used to detect and recall an accident as it happened. This would require the smartphone running the application to be in the vehicle at the time of the accident. During their research, Singh et al. (2013) noted that at least 1.4 million people died due to road accidents in India in 2011 [16]. In addition to that, one of the leading causes of death in the US are car accidents with 10.6 million traffic accidents being reported in 2007 [18]. This shows that road accidents occur at a high rate in both developed and developing regions. While sometimes accidents cannot be avoided, the long response time taken by emergency services to arrive at the scene increases on the number of fatalities in serious accidents [3].

A variety of services have been developed to keep road users safe on the road. One such service is OnStar which provides automatic car crash response as well as diagnostics for vehicle maintenance, among other services [1]. The eCall project, started by the European Union, requires that by 2018, an automatic accident notification system has to be built in each new car [9]. While such systems are certainly better equipped than smartphones, they are expensive and only available in new cars [9]. According to White et al. (2011), in 2007 most cars in the US did not have automatic detection and notification systems [18].

Smartphone applications can and have been developed to detect accidents and notify emergency responders. In doing their research, White et al. (2011) found that in 2010, 325.6 million devices were sold in the second quarter [18]. Research done by Amanda Lenhart (2009) showed that more teenagers own mobile phones today than ever [11]. Furthermore, the teenage demographic is historically the most accident prone age group [12]. The recent increase in the processing power of smartphones and their ubiquitous use around the world also supports the notion that smartphones can be used to research new areas [8]. This, coupled with the findings of Lenhart (2009), shows that smartphone applications that detect accidents can be developed and would be very useful for drivers.

The paper critically discusses and evaluates smartphone applications that detect car accidents and applications that monitor other traffic conditions. Critical analysis is be done on each of the applications to find the similarities and differences between the different implementations along a number of dimensions such as the architecture used.

Section 2 gives an overview of the hardware and software in mobile devices that is used in the development of these applications while section 3 gives an overview of the applications. A comparison (and analysis) along three dimensions namely; approach, architecture and algorithms is conducted in section 4. Section 5 presents a critical analysis and evaluation of the accident detection applications. The paper ends with a conclusion in section 6.

2. MEASUREMENT SYSTEMS ON MOBILE DEVICES

This section gives an overview of the various hardware and software in mobile devices that can be used to develop applications to monitor traffic. Section 2.1 looks at sensors while section 2.2 mentions some of the other systems in mobile devices that have been used.

2.1 Sensors

One of the main components of smartphones that is vital to detecting accidents is the sensor(s). Karin et al. (2005) stated that in order for a mobile phone to interact with the real world, a sensor in the device must be used to establish a connection between the two [10]. Essentially, a sensor is an object used to detect changes or occurrences in its surroundings. A common example of sensors is seen in automatic doors which detect the presence of an object in close proximity and causes them to react accordingly.

Karin et al. (2005) observed that mobile phones had three built-in sensors [10]. These were a camera, microphone and a network interface. Like most technology, the sensors in mobile devices have evolved. Moore et al. (2011) mentioned that the varied and powerful sensors in mobile devices such as audio sensors (microphone), acceleration sensors (accelerometer) among others could enable smartphones to do more than the basic functionalities [8]. As of today, Android developers are exposed to three broad categories of sensors that they can use namely; motion sensors, environmental sensors and position sensors [2]. A short description of each of these categories is shown in table 1. Sensor availability varies from device to device as well as between Android versions [2]. Essentially, there are many sensors that developers can use for a variety of purposes.

2.2 Other systems

Besides sensors, many of these applications use Global Positioning System (GPS) on the device. White et al.(2011) mentioned the availability of a GPS in the iPhone 4 which is used by their application [18]. This is a vital component of a mobile phone as it can identify the current location of the device. One of the drawbacks of using the GPS system is that it uses a lot of the smartphone energy, as this paper explains later [12].

3. APPLICATIONS

A number of applications and algorithms have been developed to help smartphones detect accidents and monitor traffic. This section gives an overview of these applications and how the systems mentioned above, such as the accelerometer, are used. Section 3.1 discusses the accident detection applications while section 3.2 discusses those that monitor other traffic conditions.

3.1 Accident detection

The applications described below are used to specifically detect road accidents using smartphones. Four different applications are discussed in this section. The first application is the WreckWatch application developed by White et al.(2011) which is a fully functioning application and available on the Google play store [18]. The second is the iBump application proposed by Aloul et al.(2014) [3]. Another approach is the Poster application developed by Nicholas et al. (2014) [4]. Lastly, is Car Crash Detection on Smartphones (CCDS) done by and Lahn et al. (2015) respectively [9].

3.1.1 WreckWatch

Developed by White et al. (2011), WreckWatch is an Android based application with a client/server architecture [18]. The application records the path, speed and acceleration of a vehicle leading up to and during an accident. By sampling the accelerometer in the device, the accident detection model used by the WreckWatch application is able to predict when collisions have occurred [18]. Besides accident detection, WreckWatch provides notification services to emergency contacts through the server side of the application.

3.1.2 iBump

Similar to the WreckWatch application, the iBump application by Aloul et al. (2014) is also designed a client/server architecture [3]. Using the built-in accelerometer on a smartphone, the iBump application continuously gathers and evaluates data [3]. Like the WreckWatch application, the iBump application also provides notification to emergency contacts as soon as an accident is detected. This is done through the SMS service.

3.1.3 Car Crash Detection on smartphones (CCDS)

Lahn et al.(2015) identified three reliable and general criteria which if reliably met could be used to detect any kind of car crash [9]. The criteria are:

1. The smartphone is currently inside of a vehicle.

Table 1: The types of sensors in android devices.

Sensor Type	Description or Use
Motion sensors	These sensors measure the movement of a device in various ways such as the accelerometer
Environmental sensors	Measure different aspects of the environment e.g. temperature and pressure.
Position sensors	There are two position sensors and they are used to determine the position of a device.

Table 2: Shows the main systems used and their purpose.

Component or sensor	Description or use
Camera	The camera is used to take images and or record as the car is moving.
Microphone	Used for detecting various noises in traffic such as honking.
Accelerometer	Measures the acceleration force applied to a device. One of the most commonly used sensors.
Gyroscope	Is a motion sensor that is used to measure the rotation of a device along three axes.

2. The smartphone experiences a high acceleration supposedly during the car crash.

3. The vehicle comes to a stop.

Similar to the applications mentioned above, the CCDS system gathers data from the built-in accelerometer [9]. While White et al. (2011) [18] used noise detection to help reduce the number false positives, the CCDS system ignored the noise factor but rather claimed that dropping the smartphone has a distinct acceleration pattern which can be reliably filtered out of the data stream [9].

3.1.4 Poster

Nicholas et al. (2014) designed an approach that consists of three portable devices namely; a smartphone, Raspberry Pi and SensorTags [4]. The application, developed to run on the smartphone, continuously monitors the accelerometer and gyroscope in the smartphone [4]. In order to improve the accuracy of the algorithm and reduce the number of false positives, this approach introduced a Raspberry Pi as well as SensorTags [4]. Section 5 provides more explanation on this.

This system differs from the other three applications along two dimensions. Firstly, is the use of the gyroscope to detect the orientation of the device and the vehicle. Secondly, is the use of a second pair of sensors to get a second opinion and detect accidents the smartphone sensors could have missed.

3.2 Monitoring traffic conditions

Some researchers have developed applications that are used to monitor traffic and discover any anomalies that are not necessarily road accidents. These applications monitor poor road conditions such as the presence of potholes, or driving patterns. While these applications and research are not directly linked with our project, there are different components or approaches that could provide valuable insight to our project.

Some of the prior work that has been done with regard to monitoring road conditions involves deploying sensors on the road-side [16]. A case in point is the work done by Raman et al. (2010) which deployed audio sensors on roadsides in India to gather data and detect various traffic conditions [15]. Another approach mentioned by Singh et al. (2013) involved using the senors embedded in the vehicles to detect the road conditions [16]. However, as already mentioned, not all drivers drive the latest versions of vehicles which have the sensors and the technology to do this [18]. Singh et al. (2013) added that one of the advantages of using a smartphone application to monitor traffic conditions is that it could assist the driver become more careful [16]. This section gives an overview and analysis of three smartphone applications that have been developed to monitor traffic conditions.

3.2.1 Using Mobile Phone Sensors to Detect Driving Behaviour

Singh et al. (2013) developed an application on the Android platform to monitor the driving pattern of the user [16]. This application is divided into two sections namely; data collection and data analysis. The application collects data from the accelerometer as well as audio data from the built-in microphone [16]. Analysis of this data enabled them to find a variety of patterns. In addition to finding patterns, Singh et al. (2013) stated that they could correlate the accelerometer data with the audio data to make other deductions [16].

The idea of data correlation could be useful in our project as we could need different criteria to confirm that an accident has indeed occurred. Correlation of data could be one way of reducing on the number of false positives. In their closing remarks, Singh et al. (2013) mentioned the use of machine learning algorithms to classify the data collected [16]. We agree with this idea and think that there two ways this can be approached. One way would entail using 'supervised learning' to teach the algorithm which patterns could be dangerous. Another way would involve using clustering techniques to classify the data along different dimensions like location of occurrence or time of occurrence. Such statistics could be used to find out the causes of road accidents and when or where they are likely to occur.

In their research, Mohan et al. (2008) claimed that there is a lot of aimless honking by some drivers [12]. In such cases, the data collected by the microphone could lead to wrong inferences being made and we think this is one of the major drawbacks of this application. This is also a problem in applications that use noise data in the detection of car accidents.

3.2.2 Nericell

Raman et al. (2010) stated that traffic conditions in developed and developing regions are fundamentally different [15]. In addition to this, Mohan et al. (2008) cited the varied traffic conditions in developing regions as one of the reasons for doing their research [12]. This led to the development of Nericell, an application used to monitor road conditions as well as other traffic conditions such as congestion [12]. Similar to a majority of the applications already discussed such as WreckWatch [18], Nericell uses the accelerometer to collect data for a variety of purposes. One of these purposes is detecting brake events. They concluded that the frequency of braking could be indicative of the quality of a drive [12]. Mohan et al. (2008) the used the accelerometer for purpose of detecting break events, because while GPS could provide a similar functionality, it would be at a high energy cost [12]. This is an example of how optimisation can be done in such applications.

Similar to the WreckWatch application, Nericell also uses the built-in microphone to detect any noises such as honking [12]. At this point, we notice that some of the mobile phone components such as the microphone and accelerometer are being repeatedly used for similar functions. This gives a novice researcher in this field a direction to follow.

Lastly, Mohan et al. (2008) used the concept of triggered sensing. Essentially, triggered sensing is using one sensor which low energy and less accurate to trigger the operation of a high energy and more accurate sensor [12]. An example of this is shown in the Nericell application where the GSM-based localisation triggers the GPS which provides a more accurate location [12]. This ensures that energy cost effective sensors are used for most of the time and the expensive sensors are only used when accuracy is desired.

The major take-away from this application is that different sensors consume the smartphone battery at different rates. Therefore, it is imperative that developers research on this and use the sensors that consume the least battery while providing the best service.

3.2.3 Mobile Assistant for Inattentive Drivers

Tong et al. (2012), discussed an approach that uses the smartphone camera to watch the driver's face and watch the road as well [17]. In our research, this is the first approach that makes use of the camera and is similar to want we want to implement for this project. This approach could enable recording of any collisions as they occur. Tong et al. (2012) claimed that with a combination of image processing and data from other sensors such as audio data, a smartphone could be used to approximate the safety features in luxury cars [17].

This approach has a few drawbacks. One of them is the fact that the application has to continuously switch between the front and back camera. This leads to latency, as Tong et al. (2012) found out during their research [17]. Although this paper claims they can overcome all the challenges they had thought of, one issue still stands out in our opinion. We cannot be certain that something vital will not be missed while the application is switching from one camera to another. We think users can also miss something important on one of the cameras when the other is being used.

However, the use of the "camera sensor" can be used to reduce the number of false positives or rather help to confirm that an accident has indeed occurred. An accident detection application could run in the background while using the camera to record what is happening. Upon detecting an accident, the application would then send a short video to the servers which would then be viewed to confirm the accident. This would require fast and efficient aggregation of the data.

4. DESIGN COMPARISON

This section compares the four accident detection applications along three dimensions namely; architecture, algorithms and approach.

4.1 Architecture

The WreckWatch application is Android based application with a client/server architecture [18]. This is similar to the work done by Aloul et al. (2014) in the development of the iBump application [3]. The iBump system consists of an application which runs on a smartphone and an application server. While the application uses the sensors on the mobile device to detect accidents, the server provides a variety of other services [3]. The application server is used for reporting accidents as they occur as well as registering and tracking users [3].

Based on this, we can conclude that to develop a successful accident application we need to take into consideration the client, server and the effective communication between the two, especially for real-time reporting of accidents.

4.2 Approach taken

As already mentioned, the Poster prototype consists of three portable devices namely; a smartphone, Raspberry Pi and SensorTags [4]. The approach taken by Nicholas et al. (2014) attempts to detect two types of accidents namely; (1) accidents where the vehicle has a quick and significant change in orientation and (2) forceful accidents such as head-on collisions [4]. This is different from the approach used by Lahn et al. (2015) who identified three reliable and general criteria (mentioned in 3.1.3) which if reliably met could be used to detect any kind of car crash [9]. The algorithm designed for this approach assumes that all the vehicles involved in a car accident experience a high acceleration before coming to an inevitable stop. However, that may not be the case all the time and therefore this algorithm could miss crashes that do not have the mentioned characteristics. On the other hand, Aloul et al.(2014) assumed that a vehicle could be in four states namely; no accident, low severity, medium severity and high severity [3]. Lastly, the WreckWatch application didn't specify any criteria but rather tried to detect any kind of collision involving the vehicle [18].

The main take away from this sub-subsection is that different approaches can be used but it is certainly difficult to find a perfect approach.

4.3 Algorithms

The Poster application attempts to detect both forceful accidents (such as head-on collisions) as well as those where a vehicle has a significant change in orientation. This is achieved by continuously monitoring the accelerometer and gyroscope in the smartphone [4]. The gyroscope is used to detect accidents where a vehicle has a quick and significant change in orientation, while the accelerometer detects forceful accidents such as head-on collisions [4].

The accelerometer is one of the common sensors used in this field, as all four of the applications reviewed use it in their detection algorithms. In addition to the accelerometer, the CCDS system also uses location sensors to determine the location of the device [9]. One unique feature only seen in the CCDS system is the use of a pipeline architecture to process the data. After obtaining the acceleration and velocity data from the accelerometer, the CCDS system uses a pipeline architecture to filter and combine the data before comparing it to a predetermined threshold [9]. As mentioned by Lahn et al. (2015), the use of a pipeline architecture reduces the complexity of the overall algorithm [9]. Furthermore, the pipeline steps can be re-used and it is easier to find any bugs in the algorithm.

The use of a predetermined threshold is also seen in the Poster application as well as the WreckWatch application. To predict if an accident had occurred, the WreckWatch application filters the data from the accelerometer based on a predetermined threshold [18]. Essentially, acceleration events are used to determine if an accident has occurred. For the Poster prototype, Nicholas et al. (2014) designed an algorithm that compares accelerometer and gyroscope data against multiple predetermined thresholds [4].

Rather than use a threshold-based method, the iBump application uses both Dynamic Time Warping (DTW) and Hidden Markov Models (HMM) to predict a collision [3]. Dynamic Time Warping is a technique used to find an optimal alignment between any two time-dependent sequences [13]. Meinard (2007) went on to explain that DTW can be used to compare patterns [13]. Meanwhile, Hidden Markov Models are mainly used for modelling linear problems such as time series [5]. Aloul et al. (2014) used DTW to differentiate between the four states (mentioned in section 4.2) [3]. After testing their algorithm, Aloul et al. (2014) concluded that their hybrid approach yielded a false positive rate of 2 percent which was better than threshold-based methods [3]. h

4.4 Summary

Based on the above subsections, I think that accident detection applications would function better with a client/server architecture. This would enable fast and efficient aggregation of data on the server side, while the client side would cover all the calculations and data collection. However, the approach used can differ as it would be hard to cover all types of accidents. Finally, the algorithm used depends on the developer but should do as much optimisation as possible. One of the main talking points not mentioned above is the prevention of false positives in such applications which is mentioned in section 5 under evaluation.

5. EVALUATION OF THE APPLICATIONS

This evaluation is mainly for the accident detection applications. However, some of the other applications are mentioned briefly where necessary. In addition to this, this section concludes with some of the challenges facing developers of such applications and what they should focus on. Huy et al. (2012) settled on an evaluation model for mobile applications that consisted of three criteria namely; developer's viewpoint, users' viewpoint and service provider viewpoint [7]. Since we neither developed nor used any of the applications, we devised three criteria to evaluate the applications. These criteria are; optimisation, false positives, data aggregation and notification and are described in sections 5.1, 5.2 and 5.3 respectively.

5.1 Optimisation

Ferriera et al. (2011) explained that the increased functionality and usage of smartphones means that devices requires more power to sustain these processes for a period of time [6]. In addition to that, smartphones have services running in the background that use up the battery. This is why it is important to design smartphone applications with optimisation of the smartphone energy at the forefront. This evaluation section reviews how the different applications tried to achieve optimisation. As well as optimisation of energy, the section explains how some algorithms were optimised to run faster.

The leading optimisation (with regard to smartphone energy) method we have come across so far is triggered sensing used in the Nericell application. This is a sound technique as it allows the application to control which sensors it uses at all times. The Nericell application uses the GPS, one of the most expensive sensors, only when it needs the accurate location of the device [12]. None of the four accident detection applications discussed in this paper used such a technique. This makes the determination of the location of a device an expensive operation because the GPS is always used. A case in point is the WreckWatch and iBump applications which constantly use the GPS to find the location of the device [18, 3]. White et al. (2011) mentioned that one of the challenges faced by smartphone applications in this field is that the application consumes a lot of battery power [18]. However, White et al.(2011) did not attempt to optimise their application with regard to energy consumption but rather suggested that users could charge their devices in their vehicles [18].

The system developed by Nicholas et al.(2014) had the same algorithm running on both a smartphone and a Raspberry Pi [4]. The Raspberry Pi is used to read and process the data from a pair of SensorTags, which is the third component of this system. With this method, the smartphone does not have to communicate(through wireless communication) with the SensorTags and thus, saves a lot of energy. Furthermore, we think that the Raspberry Pi can be used to do all the complex calculations thus, ensuring the smartphone does even less computation.

Finally, the pipeline architecture used by Lahn et al. (2015) in the CCDS system provides optimisation because it simulates parallelism [9]. Pipline computing increases the throughput of the algorithm. This is possible because of the processing power available to smartphones today. White et al.(2011) cited the HTC Nexus One smartphone which has 1GHz processor and 512 RAM [18]. Smartphones have improved and now have multi-core systems in addition to the strong processors. This provides a basis for applications to use parallel algorithms to ensure faster processing.

As evidenced above, not all developers put optimisation at the forefront during their research and development. This is one of the gaps that our project will try to address. The concept of triggered sensing can save a lot of energy, while pipe-lining could ensure that the algorithm runs faster. Both these techniques provide different types of optimisation and can be used in development of the same application.

5.2 False positives

As mentioned by White et al.(2011), one of the challenges of accident detection applications is the prevention of false positives [18]. This subsection reviews how each of the applications tried to reduce on the number of false positives.

As already discussed in detail, the iBump application unlike the other three accident detection applications does not use a threshold based method, but rather uses a combination of DTW and HMMs which Aloul et al.(2014) concluded had a better performance with regard to false positives [3]. This is certainly different from the Poster system which uses SensorTags to reduce on the number of false positives [4]. The Raspberry Pi runs the same algorithm as the smartphone and evaluates the data collected from the SensorTags [4]. Different results could signal a false positive or that one of the devices could be wrong. An improvement on this would be having two different algorithms running, to test for efficiency of the algorithm and which sensors are more accurate. The downside to this prototype is the use of SensorTags and the Raspberry Pi which cannot be carried on every journey.

In a bid to reduce on the false positives and to have the ability to detect low speed collisions in the WreckWatch application, White et al.(2011) used the built-in microphone to detect accident noises such as airbag deployment and car horns [18]. Aimless honking by some drivers can distort the findings or lead to wrong predictions [12].

Lastly, the CCDS system developed by Lahn et al.(2015), claimed that dropping a mobile device has a distinct acceleration pattern which they filtered out [9]. They went on to state that shaking a mobile device could also lead to false positives but did not cater for this.

5.3 Data aggregation and Notification

This section discusses how the data was aggregated and used. This is vital especially with accident detection where the data needs to be relayed onto emergency contacts as fast and accurately as possible.

Only the iBump and WreckWatch application provide notification services. As soon as an accident is detected, the iBump application (client side) sends an SMS to the emergency contacts and police [3]. On the other hand, the WreckWatch application uses the server (server side) to communicate to emergency responders as well as aggregate all the data collected [18]. In addition to this, the server (WreckWatch application) posts location and severity information to help the emergency responders.

We think that the WreckWatch application uses a better strategy than the iBump application because the server aggregates all the data and then relays onto the emergency responders. Not only is it a more effective process, but the server can also easily send data faster over longer distances.

Table 3 summarises the pros and cons of each of the applications reviewed.

5.4 Challenges

In researching the prior work, we discovered two main problems facing accident detection applications.

In the development of the WreckWatch application, White et al.(2011) claimed that one of the major challenges that developers of such applications had to solve is the prevention of false positives [18]. Lahn et al. (2015) suggested that a major cause of the false positives is the device being dropped, but observed that the distinct pattern of the fall could be filtered out of the

Table 3: A table showing the pros and cons of each application

Application	Pros	Cons
WreckWatch by White et al. (2011)	-Server side that does data aggregation. -The server is configured to notify emergency contacts without interacting with client thus saving time.	-Noise data used to reduce on the number of false positives is not consistent especially in traffic congestion
iBump by Aloul et al. (2014)	-DTW and HMM methods used can detect the severity of an accident. -DTW and HMM methods have less false positives than the more common threshold-based methods.	-Use of DTW and HMM could lead to computational overhead and drain the smartphone battery.
Poster by Nicholas et al. (2014)	-Uses a second set of sensors in addition to the smartphone sensors which increases on the accuracy of results. -The Raspberry Pi and Smartphone run the algorithm concurrently therefore false positives are reliably detected.	-Difficult to implement such a system with all the components.
CCDS by Lahn et al. (2015)	-Optimised the algorithm by using a pipeline architecture to evaluate the data.	-The detection and removal of false positives is not comprehensive enough.

data stream [9]. Therefore, such applications have to be able to detect the difference between an accident occurring and a smartphone being dropped to the ground.

Another challenge for such applications is reliably communicating with emergency responders in the event an accident is detected. As stated by White et al. (2011), accident detection applications running on smartphones should be able to inform first responders as soon as possible and with as much information about the accident as can be obtained [18]. More expensive systems like OnStar which has already been mentioned, can detect the seriousness of an accident using information from airbag deployment [18]. Essentially, accident detection applications should be able to communicate efficiently and effectively with first responders, otherwise it defeats the purpose of detecting the accident.

Therefore, I think such applications have two main tasks; (1) accurately detecting car accidents (with minimal false positives) and *(2) informing emergency responders as fast as possible.

6. CONCLUSION

Despite all the progress made by smartphone applications in accident detection, we am of the opinion that in-vehicles accident detection systems are better suited to detect road accidents. These systems are able to use the sensor networks in the car and interact with the vehicle's electronic control unit which a smartphone cannot do [18]. However, smartphone applications can provide a viable stop-gap solution until every driver has a vehicle with the ability to reliably detect accidents and notify emergency contacts.

The main challenge for all the applications is reliably filtering out any false positives [9]. One sure way to know whether an accident has occurred or not is to view what has happened. This would entail using the camera to constantly record as the car is moving. As soon as an accident is detected, a short video is sent to the server depicting exactly what happened. In addition to confirming accidents, the short videos can be studied to identify the causes of accidents and the common locations where they occur. The use of the camera will be an area of focus in our project.

One concept not mentioned by any of the applications reviewed is the security of the data. It is not clear whether the data on the servers is secured and backed up. Providing security features would enable the server save more data about the users that could be used in case of an emergency. This is one of the gaps in this field. Our project is focusing on mainly collision detection and recording of the accident, and therefore it is out of the scope.

An improvement can be made in the aggregation of data and notification of emergency contacts. The iBump application uses the SMS service to notify emergency contacts when an accident is detected [3]. Communication with emergency responders can be improved by using both the application (client side) and the server to send a message. The application would provide an instant notification while the server would send a more detailed message a few seconds later. This would require effective and efficient data aggregation on the server side. In addition to this, a stable and effective network would be required to transmit the data as fast as possible. One way of achieving this would be through the use of Content Delivery Network (CDN). The use of a CDN ensures that the data is sent to closest servers and is accessible as soon as the accident has occurred[14]. Pallis et al. (2006) also added that the use of a CDN reduces congestion over servers [14]. This could directly reduce on the time spent waiting for the emergency responders.

There is a lack of a visual aspect to the accident detection as well as a perfect solution to preventing false positives. This presents an opportunity for contribution by our project in the work it will be conducting.

7. REFERENCES

[1] Onstar. https://www.onstar.com/us/en/home.html. Accessed: 2016-04-20.

[2] Senors overview. http://developer.android.com/guide/topics/sensors/sensors_overview.html. Accessed: 2016-04-20.

[3] Fadi Aloul, Imran Zualkernan, Ruba Abu-Salma, Humaid Al-Ali, and May Al-Merri. ibump: Smartphone application to detect car accidents. *In Industrial Automation, Information and Communications Technology (IAICT), 2014 International Conference on (pp. 52-56). IEEE.*, 2014.

[4] Nicholas Capurso, Eric Elsken, Donnell Payne, and Liran Ma. A robust vehicular accident detection system using inexpensive portable devices. *In Proceedings of the 12th annual international conference on Mobile systems, applications, and services (pp. 367-367). ACM.*, 2014.

[5] Sean R Eddy. Hidden markov models. *Current opinion in structural biology*, 6(3):361–365, 1996.

[6] Denzil Ferreira, Anind K Dey, and Vassilis Kostakos. Understanding human-smartphone concerns: a study of battery life. *Pervasive computing*, pages 19–33, 2011.

[7] Ngu Phuc Huy and Do vanThanh. Evaluation of mobile app paradigms. In *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia*, pages 25–30. ACM, 2012.

[8] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

[9] Julia Lahn, Peter Heiko, and Peter Braun. Car crash detection on smartphones. *Proceedings of the 2nd international Workshop on Sensor-based Activity Recognition and Interaction. ACM, 2015*, 2015.

[10] Karin Leichtenstern, Alexander De Luca, and Enrico Rukzio. Analysis of built-in mobile phone sensors for supporting interactions with the real world. *In PERMID (pp. 31-34)*, 2005.

[11] Amanda Lenhart. Teens, smartphones & texting. *Pew Internet & American Life Project*, 2012.

[12] Prashanth Mohan, Venkata N, and Ramachandran Ramjee. Nericell: rich monitoring of road and traffic conditions using mobile smart-phones. *In Proceedings of the 6th ACM conference on Embedded network sensor systems (pp. 323-336). ACM.*, 2008.

[13] Meinard Müller. Dynamic time warping. *Information retrieval for music and motion*, pages 69–84, 2007.

[14] George Pallis and Athena Vakali. Insight and perspectives for content delivery networks. *Communications of the ACM*, 49(1):101–106, 2006.

[15] Rijurekha Sen, Bhaskaran Raman, and Prashima Sharma. Horn-ok-please. *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 137–150, 2010.

[16] Pushpendra Singh, Nikita Juneja, and Shruti Kapoor. Using mobile phone sensors to detect driving behavior. *Proceedings of the 3rd ACM Symposium on Computing for Development*, 2013.

[17] Sanjeev Singh, Srihari Nelakuditi, Romit Roy Choudhury, and Yang Tong. Your smartphone can watch the road and you: mobile assistant for inattentive drivers. *Proceedings of the thirteenth ACM international symposium on Mobile Ad Hoc Networking and Computing*, pages 261–262, 2012.

[18] Jules White, Chris Thompson, Hamilton Turner, Brian Dougherty, and Douglas C. Schmidt. Wreckwatch: Automatic traffic accident detection and notification with smart-phones. *Mobile Networks and Applications 16.3 (2011): 285-303.*, 2011.