DropAlert – Android Smartphone Application

ANUDEEP BODIREDDY, Master's Student in CS at Binghamton University
ANIKETH REDDY MALLESH, Master's Student in CS at Binghamton University
SAMARA SIMHA REDDY, Master's Student in CS at Binghamton University

With the increasing computational power of smartphones and extent of use, there is a tremendous surge in using these small and smart systems to deal with many problems these days. DropAlert is a Android smartphone application which monitors, computes and triggers alerts when a Fall takes place. This application is aimed at helping senior citizens who needs help when these mishaps happen and many lives could be saved with appropriate response from care takers and authorities. This paper demonstrates the application, UI screens, monitoring accelerometer values and algorithm used to decide if it's a fall and alerting mechanism. From there, we put forward the experimental environment and results to evaluate the effectiveness of this application for various types of falls. Finally, we conclude this paper with major features of this application and some insights on further developments that can occur for this application to increase the effectiveness by adopting Machine Learning techniques.

KEYWORDS

Android Programming, Accelerometer sensors, Location, Social networking and messaging services

ACM Reference format:

Ben Trovato, G.K.M. Tobin, Lars Thørväld, Lawrence P. Leipuner, Sean Fogarty, Charles Palmer, John Smith, and Julius P. Kumquat. 1997. SIG Paper in word Format. *ACM Trans. Web,* 9, 4, Article 39 (March 2010),4 pages.

DOI: 10.1145/1234

1 INTRODUCTION

As people age, the likelyhood of them being vulnerable to Falls and various mishaps increases substantially. Falls are one of the major causes of injuries and deaths among people especially senior citizens. To back these statements by statistics, here are some of the statistics that asserts the statements made. Every year, more than 11 million people fall [2]. In 2005, unintentional falls accounted for an estimated 56,423 hospitalizations and 7,946 related deaths in the United States [3]. Many of these deaths are a result of a "long-lie," an extended period of time where the victim remains immobile on the ground [4]. Just the simple fear of a long-lie or falling can lead to one's lower mental health, isolation, and general degradation of quality of living [5].

Another use case for this application would be fatal crashes. Whenever we are going at high speed and crash into something, the velocity is suddenly dropped and we can use the same application to trigger alerts to authorities and rescuers that an accident has happened and would greatly decrease the response times.

Although there are some systems that have been developed in the recent past to deal with these mishaps like Image recognition and worn device systems, our solutiion of using smart phones as a replacement of these forementioned systems greatly reduces cost and and increases the extent to which we can monitor people. The forementioned systems needs people to buy some worn devices which incurs additional cost and also overhead in maintaining such Image recognition systems with

cameras installed. Also, it is not possible to cover every location with cameras for obvious reasons. As everyone uses smartphone and particularly Android smartphones, our solution is more reachable to users and very cost effective as there is no additional overhead costs associated. We leverage the computational power of smart phones equipped with a variety of sensors to detect the falls. Following sections describe the application, features, Algorithms used and alerting mechanism. Then, we evaluate the application with statistical data to measure the effectiveness of the use of algorithm. Finally, we conclude the paper with major features and uses of DropAlert and giving insights on further development of this application to increase the effectiveness to 100%.

---

2 SOFTWARE & FEATURES

2.1 Software

We used Android development stack for this application. Firebase is used to analyze the data of Falls to help drive our effectiveness in designing the algorithm. We studied the accelerometer values for various movements of phone and plotted graphs to analyze the same. We used sensor logger, an application that senses the data and plot graphs for the data given. We used, Andrioid studio and used location, messaging and social networking services to get the location data accurately, creating alerts to rescuers and authorities and to share the false positives i.e., in the event where a user did not fall but only his/her phone fell in social networking sites to say that they fell their phone again and again. The monitoring activity continuously runs in the background and only woken up when a Fall takes place as computer by our algorithms. Application takes permissions to use Contacts, GPS services and Social networking services to have access to them.

2.2 Features

In this sections, we present the UI screens of the application explaining the capabilities of the applciation. Firstly, the application is simple to use and user has the option of adding the rescuers information who they live to be notified when a mishap takes place. Also, for ease of use, we made the contacts easy by incorporating the auto-complete feature. As user types a contact's phone number or name, he/she gets the suggestions to select and add them with ease.

Also, user has the capability of calibrating their Falls, we have done this to make it easier for our algorithm to expect a Fall per individual basis. Some people who are old will have less dynamic movements than young people. So, by having this option, one can tell what a Fall is for them and algorithm will take the input once provided by the user to decide if a Fall took place. Even if user doesn't calibrate, then we have some default values to use which we came up with, when we were simulating various falls. We take 5 simulations in total for calibrations and user can fall with different postures to make the algorithm understand about possible different posture falls. User has to click "Calibrate" and then fall to make the application learn about his/her falls. User has to do it 5 times one each for a calibration screen.

Once, the calibration is done, user can "Switch On" the DropAlert service to start monitoring the application for falls. Sometimes as user doesn't want to get notified of possible falls, for example, when a user in an adventure park, he sure knows that he/she Falls and these triggers of a Fall can

create chaos. In such cases, user can “Switch off” the monitoring service for that period of time when the application stopes monitoring the accelerometer values and it can save the battery life as well.

Once a Fall takes place, user is presented with a ringtone, whichever is the default ringtone for that piece of smart phone and a “Cancel” button. The reason why we have this is, when a user Falls and if he/she feels OK and doesn’t need any attention from rescuers, they can hit the “Cancel” alert which will void the FALL detected and will not set any triggers for alerting the rescuers. It will also help in dealing with some of the false positives.

If the user hits “Cancel” and the event in which his/her phone fell again, the user can directly post this to any social networking application installed in the phone like messages, facebook chats, Facebook status, WhatsApp status etc., to let their dear ones know that they broke their phone.

As described above, the application is very simple to use and user friendly with large buttons of size to make it easier for users. In the next section, we present the algorithm developed and we follow it with the results of the evaluation of the application to measure its effectiveness.

3 Algorithm

3.1 Two Modes

There are two modes of the algorithm,

- Real Time Processing
- Delayed Processing

For both modes we read the ax, ay, az values from the accelerometer. Real Time Processing mode runs as backend android service, stores data locally and detects fall within 10 secs window. Whereas, delayed processing is based on both real time and offline processing where the calibration data is stored\retrieved from Firebase data base. Android supports different time delays for reading sensor.

- SENSOR_DELAY_NORMAL - 200,000 us delay
- SENSOR_DELAY_GAME - 20,000 us delay
- SENSOR_DELAY_UI - 60,000 us delay
- SENSOR_DELAY_FASTEST - 0 us delay

We have used SENSOR_DELAY_GAME. Then we compute the vector sum A of the ax, ay, az. This is used by threshold based algorithm for deciding the fall.

The fall detection algorithm searches for particular pattern in the signal. Figure 1 represents a typical pattern in vector sum A during a fall event

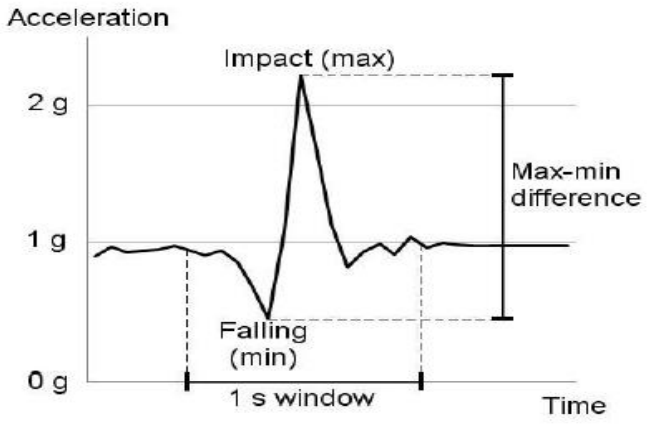


Figure 1: Typical Fall Pattern

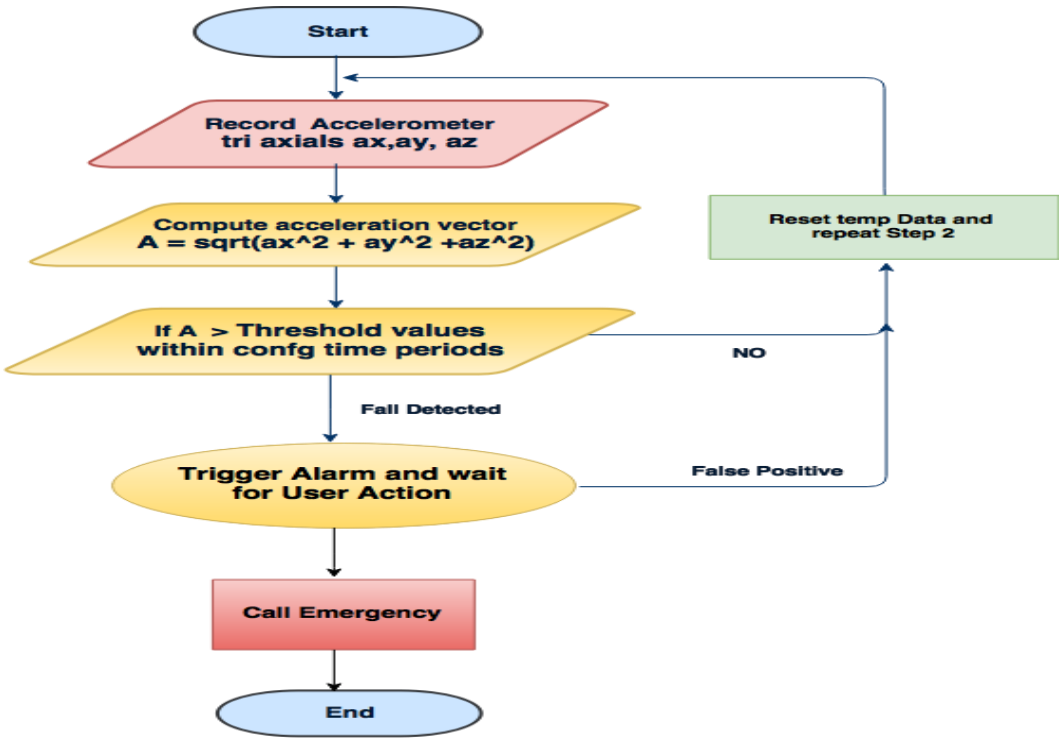
3.2 Implementation of the Algorithm

When a fall event occurs, there is a sudden drop in vector sum A, followed by a sharp rise before settling down to typical value of g = 9 m/s or any value between the minimum and maximum value. Our algorithm checks for sudden drop in vector sum against a minimum threshold ranging between 0.1 g - 0.3g, if vector sum breaks the minimum threshold, the algorithm clocks the timestamp t1 and then looks if the sharp rise in vector sum A breaks maximum threshold values 2.7 g, if yes it clocks second timestamp2 and checks if the time difference between t1-t2 is between 200ms-400ms. If yes, it triggers the alert notification and waits for user to take any action. If no action is taken, the app sends emergency message with GPS coordinates to registered contacts for help. Also, the current application allows you the user to use other social messenger apps like WhatsApp, FB Messenger, Hangouts etc. for sending you GPS coordinates. If any of the above check conditions are not met, the algorithm resets temporarily stored timestamp, vector sum values and starts over again.

3.3 Calibration of Threshold Values

Current implementation of the application as default values for the thresholds i.e. minimum/maximum and time difference values. Also, allows the user to calibrate his own threshold values for better accuracy. The more simulation a user does the better the detection.

Flow Chart of the Algorithm:



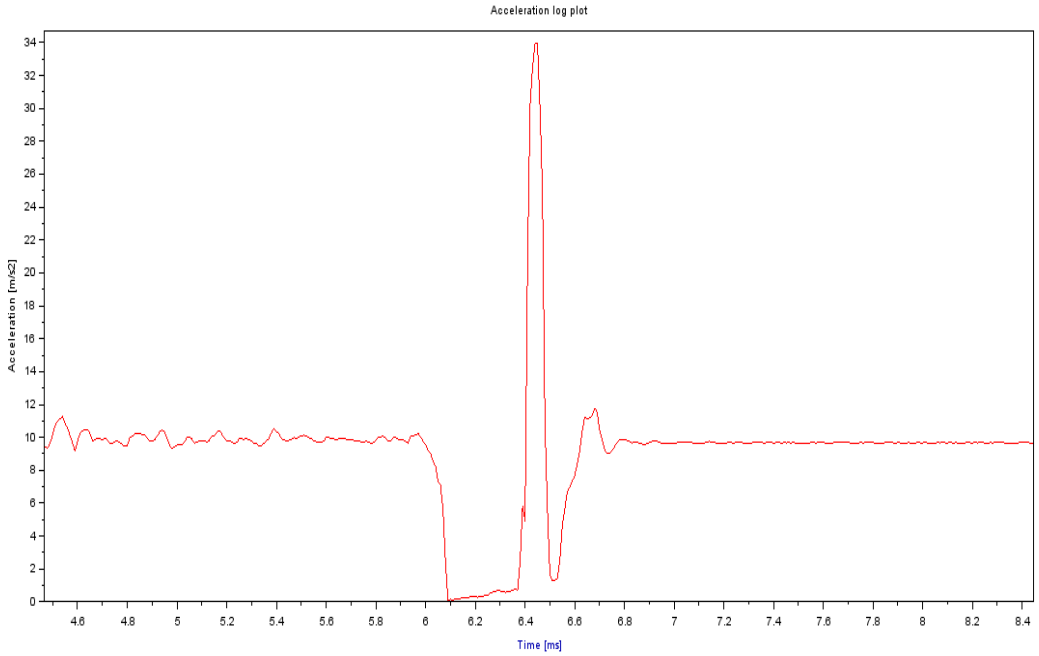


Figure 2.1: Pattern of a real backward fall, figure 2.2 is snapshot captured in firebase for this pattern.

1	Time (s)	Accel-magnitude (m/s^2)
392	6.350148	0.723108
393	6.360133	0.762116
394	6.370166	0.733617
395	6.380139	2.445505
396	6.390142	5.807706
397	6.400151	4.926687
398	6.410122	20.548094
399	6.420177	30.139204
400	6.430201	32.460239
401	6.440124	33.93771
402	6.450149	33.93771
403	6.460162	30.095869
404	6.470144	24.979856
405	6.480129	10.931955
406	6.49013	5.610296
407	6.500146	1.647779
408	6.510137	1.272254
409	6.520145	1.332277
410	6.530171	1.435961
411	6.540127	2.535938
412	6.55015	4.653751
413	6.560137	5.677239

Figure 2.2: Data captured on Firebase, a) Green colored data represents the drop, b) Orange colored data represents the sharp rise c) Blue colored data represents the device settling down to typical g = 9.8m/s

4 Services Employed and Implementation

4.1 Messaging & Location Services

When the application detects the fall I.e. the computation of the accelerometer is greater than the threshold value recorded an alarm is triggered and alert window pops up. This alert window has two outputs when the user doesn't cancel the alert in given time an SMS alert is sent to the Stakeholders enclosing the geographic details of the location where the fall recorded.

Android SmsManager manages SMS operations such as sending data, text, and pdu SMS messages. Android location data class represents geographic location. Using the location service longitude and latitude details are fetched and embedded in the text message and sent through SMS manager service.

API'S used:
android.telephony.SmsManager
android.location.Location

4.2 Social Networking services

In case user detects as false positive, the application provides user to share the fall status and location to other social media applications.

The most straightforward and common use of the ACTION_SEND action is sending text content from one activity to another. For example, the built-in Browser app can share the URL of the currently-displayed page as text with any application. This is useful for sharing an article or website with friends via email or social networking. A URL with the given latitude and longitude details is formed and the status is sent to the application the user wishes to

5 Evaluation

5.1 Evaluation mechanism

Total 6 candidates participated in the experiment with height ranging from 5.5 to 6 feet.

Each candidate participates in 3 types of falls i.e. Backward fall, Front fall, and Side fall on a mattress with all safety measures in place. For experiment we have focused primarily on two positions, chest pocket and thigh pocket for placing the phone.

Phone models used for the experiment:

- Samsung Galaxy Nexus (Android 4.0.2)
- Sony Xperia Z3 D6653 (Android 6.0.1)
- Moto G Play (Android 6.0.1)

5.2 Statistics and False Negatives:

Before testing the application, for everyone we calibrated the threshold values with different combinations. i.e. just simulating one of the 3 types of fall separately (individual calibration) or simulation for all 3 types together (combined calibration) and conducted the tests. Below are the observations for each type of Fall. Firstly, for backward falls which are most frequent types of falls, the detection accuracy is above 85% for individual calibration and 70% accuracy for combined calibration. Secondly, the Forward Falls the observed accuracy is between 70-80% for individual calibration and 50 – 70% for combined. And thirdly, the side falls which are difficult to detect as there is more movement after a fall occurs compared to other two falls was observed to between 70-80% for individual and 50-60% for combined. Free falls of the phone on the couch always showed good results without any false negatives.

Type of Fall	Individual Calibration	Combined Calibration
Backward Falls	>85%	70%
Forward Falls	70 – 80 %	50 – 70%
Side Falls	70 – 80 %	50 – 60%
Free Falls on Couch	> 95%	> 95%

5.3 Battery Power and False Positives

Our primary focus was on improving the detection of different falls and reduce false positives, we dint not focus much on battery consumption of the application services.

We observed the battery consumption and false positives, in two time windwos – 3 hour and 6 hours, where the candidates carried the app enabled smart phones during their routine class schedule involving both in bus and on campus commutes. The observations are as follows, 2-3 false positives detected in each 3-6 hour windows. Majority of them while travelling in campus shuttles.

In delayed processing mode where the service continuously sends the data to firebase the fully charged battery lasted below 2 hours and during real time processing the battery lasted between 4-6 hours.

6 Conclusions

With a simple threshold based algorithm, we were able to achieve detection of 70% accuracy in detecting different falls. With the help of advance machine learning techniques like logistic regression analysis we can classify such falls with higher accuracy. Implementing proven fall detection algorithms makes the system highly reliable. Reliability and reduced number of false positives means greater adoption by emergency services. The importance of the cell phone in everyday life decreases the chances of being forgotten. Everyday interaction with the phone makes the interface more familiar to the user.

ACKNOWLEDGMENTS

I would like to thank Prof. Mo Sha, Professor at Binghamton University and Di Mu, Teaching Assisatant for Spring 2017 for helping us althrough this course and giving this wonderful opportunity to get valuable insights into IOT, WSN and current research going on in these fields.

REFERENCES

[1] G. I. o. T. Sauvik Das, U. o. H. LaToya Green, U. o. P. R. M. Beatrice Perez and F. W. O. C. o. E. Michael Murphy, "Detecting User Activities using the Accelerometer on Android Smartphones," July 30, 2010.

[2] Garret Brown. An accelerometer based fall detector: Development, experimentation, and analysis. Technical report, July 2005. EECS/SUPERB.

[3] K.E. Thomas, J.A. Stevens, K. Sarmiento, and M.M.Ward. Fall-related traumatic brain injury deaths and hospitalizations among older adults- united state, 2005. Journal of Saftey Research, (39):269–272, May 2008.

[4] A.K. Bourke, J.V. O'Brien, and G.M. Lyons. Evaluation of a thresholdbased tri-axial accelerometer. (26):194–199, September 2006.

[5] EW Peterson, CC Cho, and ML Finlayson. Fear of falling and associated activity curtailment among middle aged and older adults with multiple sclerosis. (13):1168–1175, 2007. Multiple Sclerosis.