FALL DETECTTION IN IPHONE

BY

Rabia Mohammad (86)

Marriam Iqbal (75)

Advisor: Dr. Aimal Tariq Rextin

Co-advisor: Shehryar Khan

Department of Electrical Engineering

School of Electrical Engineering & Computer Science

National University of Sciences & Technology

Islamabad, Pakistan

2011

FALL DETECTTION IN IPHONE

Rabia Mohammad (86)

Marriam Iqbal (75)

Advisor:

Dr. Aimal Tariq Rextin

Co-advisor:

Shehryar Khan

CERTIFICATE OF APPROVAL

It is certified that the contents and form of thesis entitled **"Fall Detection in IPhone"** submitted by Marriam Iqbal (2007-NUST-BICSE-75) and Rabia Mohammad (2007-NUST-BICSE-86) have been found satisfactory for the requirement of the degree.

Advisor: Dr Aimal Rextin

(✓) Aimal Rextin

Co-Advisor: S Ilyas.

(✓)

Dedication

Dedicated to our parents, who taught us that the best kind of knowledge to have is that which is learned for its own sake. It is also dedicated to our teachers, who taught us that even the largest task can be accomplished if it is done one step at a time.

Acknowledgement

First of all we would like to thank almighty Allah for all His blessings.

We would like to thank our parents for it was because of their prayers and support that we were able to complete this task.

Dr. Aimal Tariq Rextin has been an ideal supervisor. His sage advice, insightful criticism, and patient encouragement aided the writing of this report in innumerable ways. We would also like to thank our co-advisor Shehryar Khan whose steadfast support of this project was greatly needed and deeply appreciated.

And last but not the least: our friends; Abdullah Intekhab, Asad Nasim, Muneeb Ali, Azka Zafar Rana, Nauman Zakoor, Umer Azmat Raja and Nauman Shah who encouraged us and supported us, helped us out of tough spots and volunteered for our "experiments".

TABLE OF CONTENTS

List of Figures

Table of Figures

List of Abbreviations

FN: false negative (fall has occurred but the device does not detect it)

FP: false positive (fall has not occurred but the device detects it)

TN: true negative (fall has not occurred and the device does not detect it)

TP: true positive (fall has occurred and the device detects it)

1. Introduction

1.1 Problem Statement

The goal of developing a fall detection application for iPhone is to get rid of wearable devices that are not usually preferred by the users due to their inability to detect fall over long distances and for being quite expensive.

1.2 Motivation

Fall is an event which every human being is prone to and the probability is greater in case of older people .Statistics show that one out of three adults, of age 65 or greater, falls at least once a year (1).We cannot prevent all falls and this limitation is leading to the increase in number of falls.

Fall doesn't only physically damage a person but it also leaves psychological effects. This problem calls out for some means to notify the emergency response teams in a prompt manner so that proper arrangements can be made for the people involved.

A huge number of companies have developed specialized ***fall detection devices*** (2) .These devices are worn by the user and send an emergency signal to the nearest emergency service. However, these specialized systems have their downsides the most significant of which is high cost.

To deal with the problems faced by these devices we can use smart phones to detect fall .Smart phones, because of their ability to run complete operating system software are becoming very popular nowadays .A smart phone is a mobile phone that offers more advance computing ability and connectivity and run complete operating system software providing a platform for application developers (3). With the use of smart phones the idea to have both detection and communication components in a single device can be achieved. We propose to implement fall detection by using powerful processing power, accelerometer built into the iPhone.

In fact, fall detection has already been implemented in android phones (4), but to the best of our knowledge it is not available in iPhone.

2. Literature Review

2.1 Fall Detection Using Wearable Motion Detection Sensors

This approach involves the use of wearable devices to monitor the change in acceleration of the object to detect fall. These devices can be 3-axis accelerometer or gyroscope or both. The devices use *threshold algorithms* which raise the alarm when a threshold value of acceleration is reached.

2.1.1 Related Work

2.1.1.1 Detecting Human Falls with a 3-Axis Digital Accelerometer

Ning jia (5) proposes to use ADXL345 (a 3-axis accelerometer) for fall detection. In ADXL345 there are two interrupt pins with eight interrupt functions available including: DATA_READY, SINGLE_TAP, DOUBLE_TAP, ACTIVITY, INACTIVITY, FREEFALL, WATERMARK and OVERRUN. During a fall four important features are observed that can lead to fall detection. At the start of every fall event WEIGHTLESSNESS is observed which is even greater in case of a freefall and is detected using FREEFALL interrupt of the ADXL345. The acceleration during this period will be less than 1g. After WEIGHTLESSNESS the body hit the ground or any other object causing an IMPACT. The acceleration during IMPACT is quite high usually represented as a shock in a graph. It is detected by the ACTIVITY interrupt. The next stage of fall is MOTIONLESSNESS because the body after experiencing a fall cannot rise without delay. The INACTIVITY interrupt detects MOTIONLESSNESS. The final feature in determining the fall is INITIAL STATUS. It is compared with the sampling data and if the difference is greater than a certain threshold a fall is detected. All these features combine to form a fall detection algorithm given in the flow chart below:

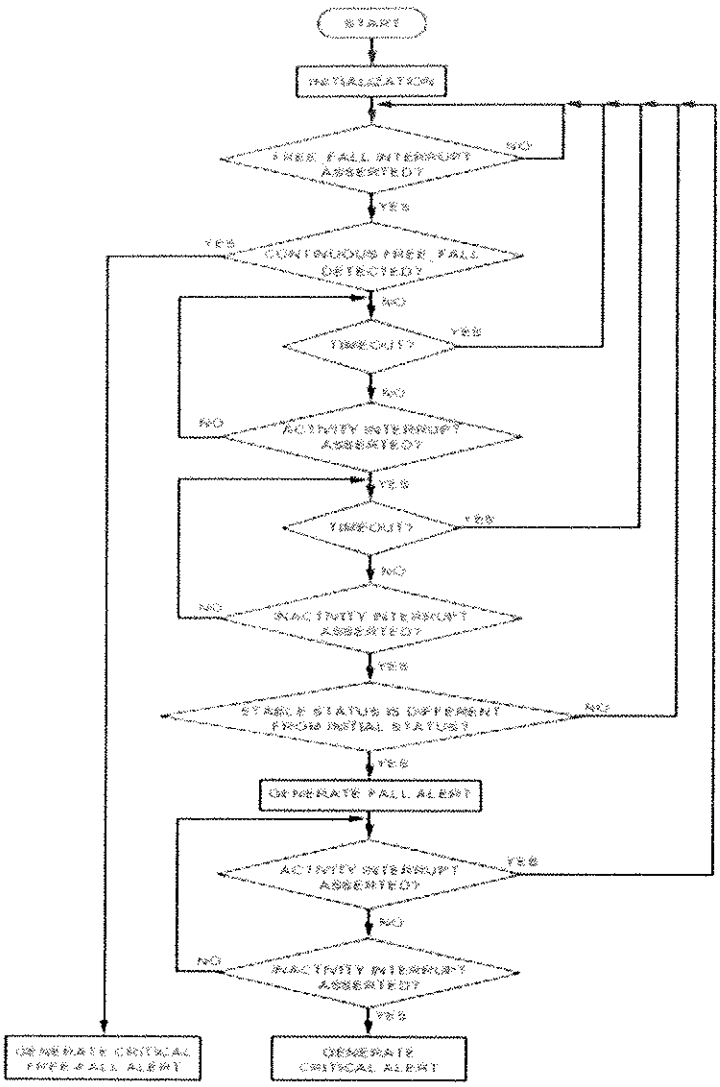

Figure 1 Algorithm

2.1.1.2 Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information

In (6) a wearable fall detection device is used that measures acceleration and angular velocity to detect falls. For this purpose TEMPO (Technology-Enabled Medical Precision Observation) 3.0 sensor nodes is used, which includes a tri-axial accelerometer and a tri-axial gyroscope. The fall detection solution is divided into three steps; activity intensity analysis, posture analysis, and transition analysis.

2.1.1.3 AWARE – Fall detection and daily activity monitoring system

Farrukh Hijaz, Nabeel Afzal and Talal Ahmed from BEE department at SEECS proposed a kinematics sensor based approach to detect fall. The system had three modules: kinematic sensors i.e. accelerometer and gyroscope, microcontroller and GSM connectivity. They encoded an algorithm in the microcontroller that detected fall using the values of acceleration measured from the sensors.

2.1.2 Discussion and perspective

The motion based sensors have a lot of advantages .The foremost is that this technique reduces the number of false positive and with the help of an accelerometer or gyroscope the accuracy to detect fall is highly increased.

However, the solution is quite costly and the deficiencies in the system do not allow pervasive fall detection. There are a number of constraints in the systems. In most cases a base system is needed to be installed in the indoors and the sensor is attached to the body thus fall can only be detected within a certain distance. Another drawback is that these devices are not daily used objects and people can easily forget to wear them.

2.2 Fall Detection Based on Smart Phones

The smart phone based approaches use the accelerometer or gyroscope or both built within a smart phone to detect fall. These applications are able to monitor the location of the user as well with the help of GPS. Smart phones along with their computational ability provide connectivity which also makes it possible for the system to send messages or call the contacts in case of emergency without any additional cost.

2.2.1 Related work:

2.2.1.1 PerFallD: A Pervasive Fall Detection System Using Mobile Phones

In (7) mobile phones were used for pervasive fall detection for the first time. They implemented a prototype system named PerFallD on the Android G1. The basic fall detection algorithm uses an acceleration threshold to detect a fall. The threshold is measured by Receiver operating characteristic (ROC) curves drawn between false negatives (FN) and false positives (FP). Receiver operating characteristic (ROC) curves are useful for assessing the accuracy of predictions and simultaneous comparison of sensitivity and specificity of the system.

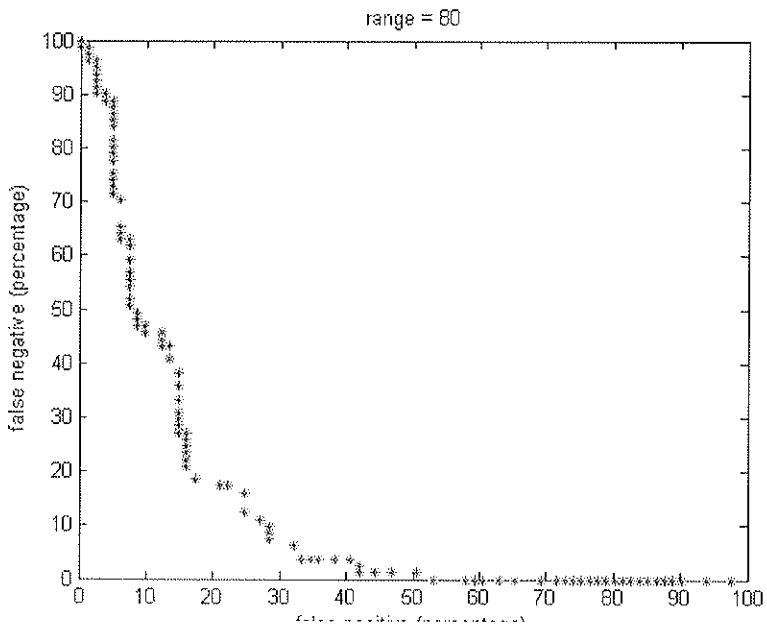

Figure 2 Relation between false negative and false positive

False negative happens when a fall occurs but the device does not detect it. False positive happens when the device alarms a fall but it did not occur. In general, the lower the both FN and FP are, the better the performance is. Initially when the program will start a user profile will be loaded which will contain the basic information such as default sampling frequency, emergency contact list etc. If the information collected in real time satisfies a certain preset condition the program will detect a fall. After the fall is detected the alert notification component will notify the emergency contacts stored earlier by the user through an alarm.

2.2.1.2 Fall Detection by Embedding an Accelerometer in Cell phone and Using KFD Algorithm

Used a mobile phone attached with a box containing a tri-axial accelerometer and MCU (micro-controller unit) and is connected to the internet via wireless channel and two set of servers. The acceleration signals are transformed into digital signals and then analyzed by the MCU. The flowchart of the main program in MCU is shown below:

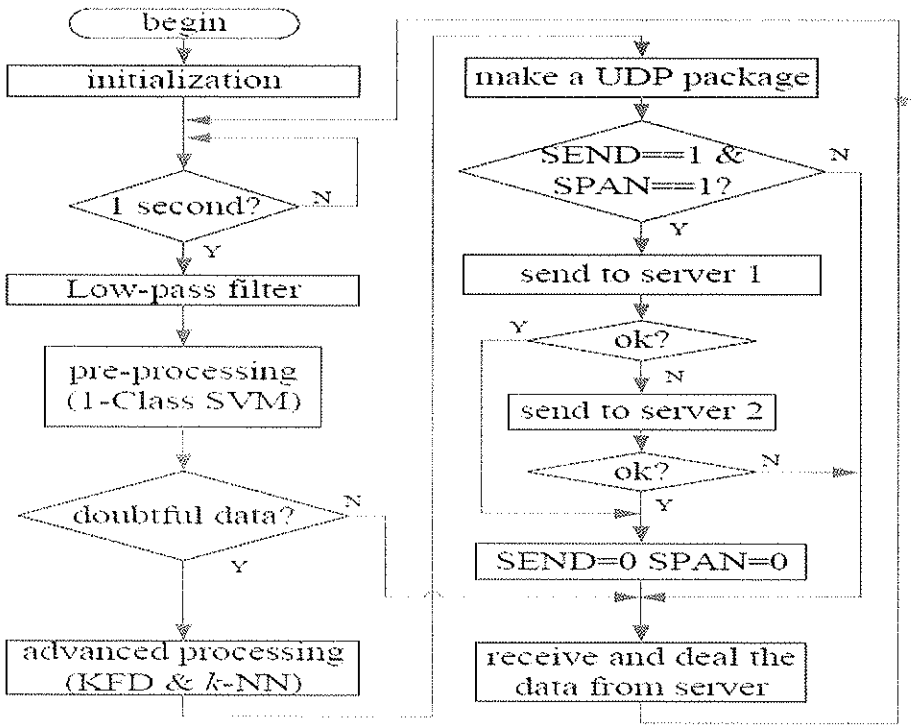

Figure 3 Flowchart of the main program

After initialization the data is first processed by SVM-based algorithm that evaluates whether the data is similar to a fall in high probability, and then the doubtful data will be classified via the KFD and *k*-NN algorithm. If fall is detected, MCU will send a UDP (User Datagram Protocol) package to a remote server via the wireless channel. At server, the sever program includes alarm display, parameter management, user interface and database management.

2.2.1.3 Fall Detection Using a Smartphone

Ian James Daniel Gonzales in (1) focuses on improving the fall detection system by highlighting the downsides of the previous fall detection systems. The approach used in this paper is to use the existing framework and add new features to overcome the faults. In their first prototype they will follow the algorithm used by (7).This algorithm uses two equations:

1: The total acceleration:

$$|AT| = \sqrt{|AX|^2 + |AY|^2 + |AZ|^2}$$

Where AT is the total acceleration of the body and AX, AY and AZ are acceleration in x, y and z-axis in reference to the body.

2: The orientation using built-in accelerometer and orientation sensor of the mobile phones.

$$|AV| = |AX\sin\Theta z + Ay\sin\Theta y - Az\cos\Theta y\cos\Theta z|$$

Where AV is the acceleration of body at absolute vertical position and θx, θy, θz are yaw, pitch, and roll values respectively.

There is an additional feature, a magnet, to reduce false positives. It will be used to measure the change in magnetic field to detect whether the fall was real or not. The users will be able to make their own profile this will help in setting threshold value specific for the type of user but for now as data available is not sufficient so the default threshold value will be used. The prototype also offers the users to set the sensitivity level of the system ranging from low to high sensitivity depending on the type of activity on a specific day.

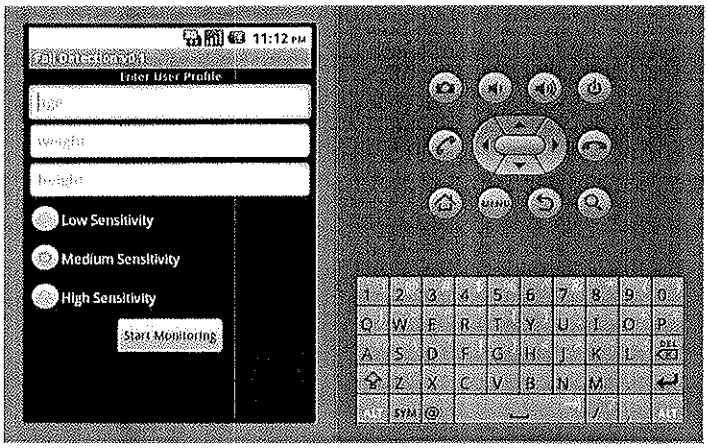

Figure 4 First Prototype

2.2.2 Discussion and perspective

The use of smart phone has resulted in making the fall detection easier by providing easier platform for application development. This approach reduces cost by using the built-in devices of the smart phones. The interface provided by these applications is also easy to use as smart phones are daily used devices and are very common. The use of smart phones has made it possible to detect fall anywhere without much constraints. The additional cost for location detection devices is also reduced as smart phones are capable of detecting location using GPS.

Like every other approach this approach also have setbacks. The major drawback of the approach is that there is no way to find out if the phone fell off accidently from the user's hand or pocket without him noticing it.

3. Methodology

The very first step of our project was to learn objective c. It was a trivial task because we were completely unfamiliar with the language. For this purpose we had to download Stanford iTune lectures.

Initially we were not assigned Mac so for practicing the basic objective c programs we used the GNUstep that works on windows. For installing the GNUstep and running our first code according to a turorial (8) we followed the following steps :

- Download
- Set environment variable for GCC compiler (C:\GNUstep\mingw\bin\gcc.exe).
- Open start -> programs -> GNUstep -> shell
- With these three steps GNUstep was successfully installed on our system.

The next steps (9) guided us on how to run our first objective c program.

- Write a simple objective c code.

```
//////////////////////hello.m//////////////////////
#import <Foundation/Foundation.h>

@interface HelloWorld : NSObject
- (void) hello;
@end

@implementation HelloWorld
- (void) hello {
    NSLog(@"hello world!");
}
@end

int main(void) {
   HelloWorld *hw = [[HelloWorld alloc]
init];
 [hw hello];
 [hw release];
}
////////////////////////////////////////////////////
```

Go to the directory where hello.m file is saved using the command:

$ cd c:/objective

Compile the program using:

$ gcc -o hello hello.m

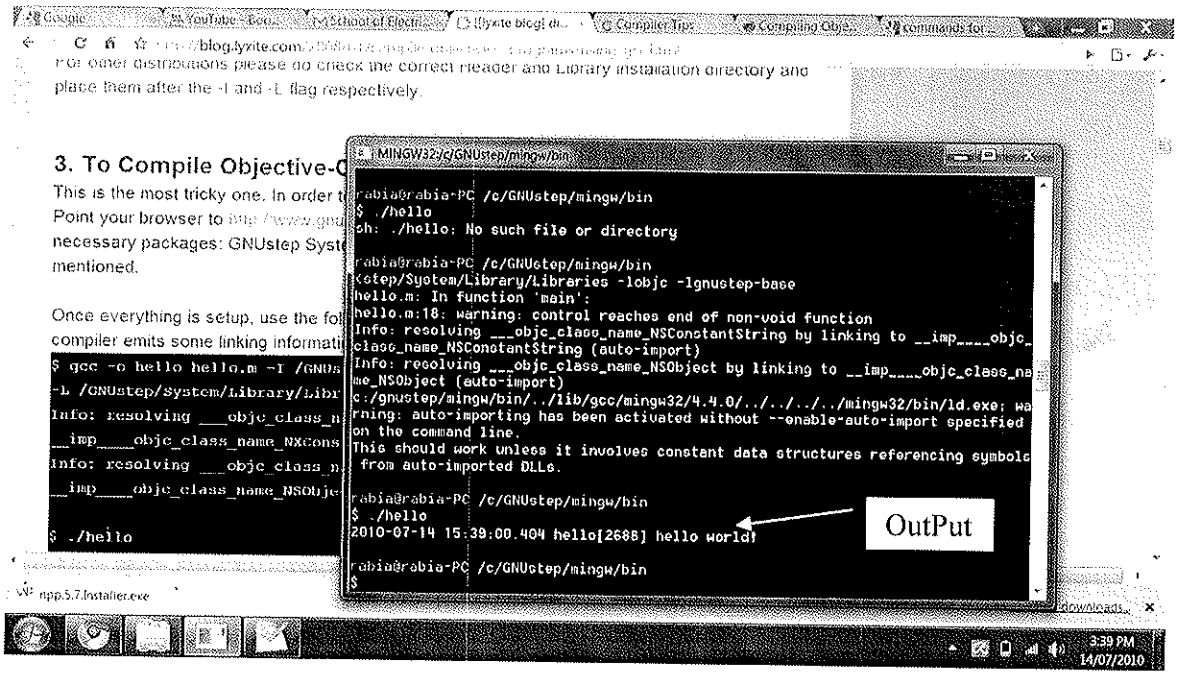

Figure 5 Hello world output

Run the program using the command:

$./hello

After following all these steps initially we didn't get the output but later with changing few settings we were able to run our first code.

Once we were assigned Mac we no longer required the GNUstep and were introduced with Xcode .As we were new to the Mac environment and xcode our next task was to learn using xcode and Mac.

Figure 6 Xcode

Xcode is a compiler and a complete toolset for building Mac OS based applications (10). Xcode handles these information while building our project (11).

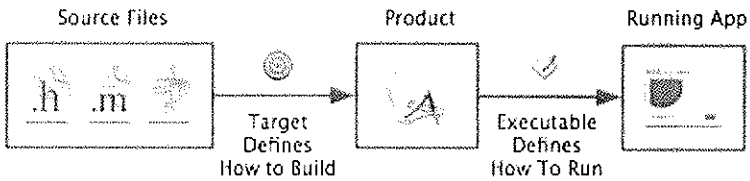

Figure 7 How Xcode uses source file references, targets, and executable environments

The interface of Xcode is quite simple there a list on left showing all the groups and files and the selected files are shown on the right. The groups and files contains the classes; for every view there are two classes; a header where all the interface elements and methods are declared and the main class where all the events are handled. The file resources in the groups and files contain the .xib file (an interface builder file).

Figure 8 Xcode components

The interface builder in Xcode helps in developing the user interface of the application by just dragging and dropping UIbutton, UIlabels and any other UI objects.

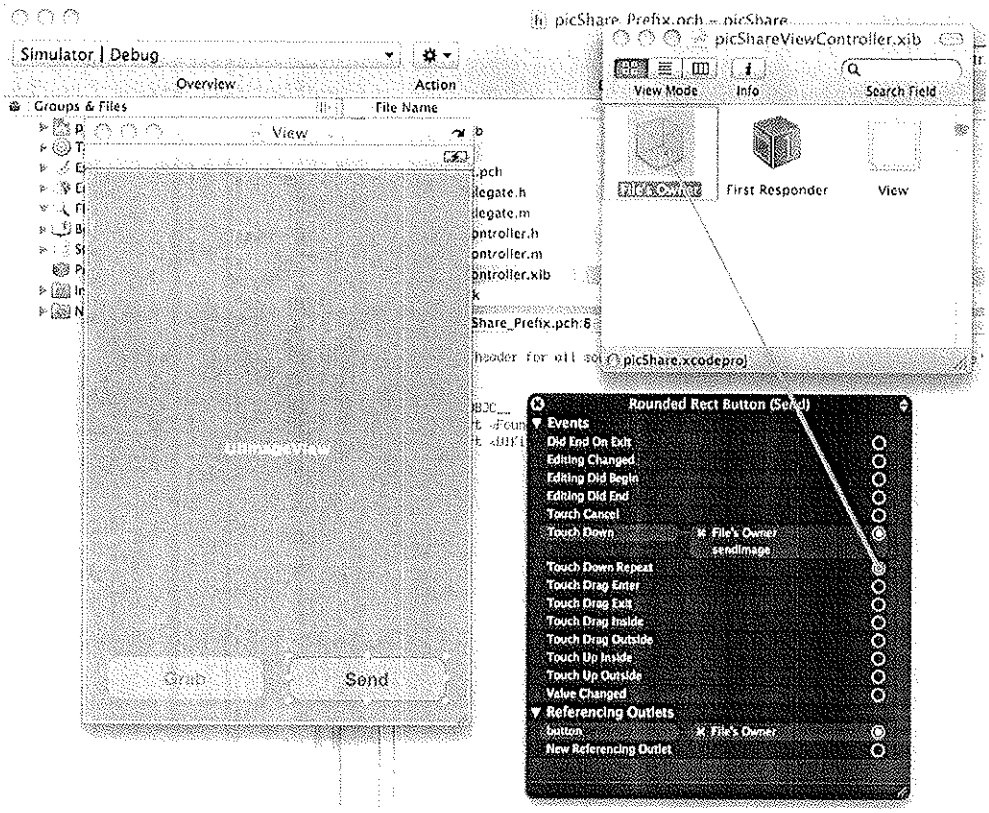

Figure 9 Xcode interface builder

Figure 8 Xcode components

The interface builder in Xcode helps in developing the user interface of the application by just dragging and dropping UIbutton, UIlabels and any other UI objects.

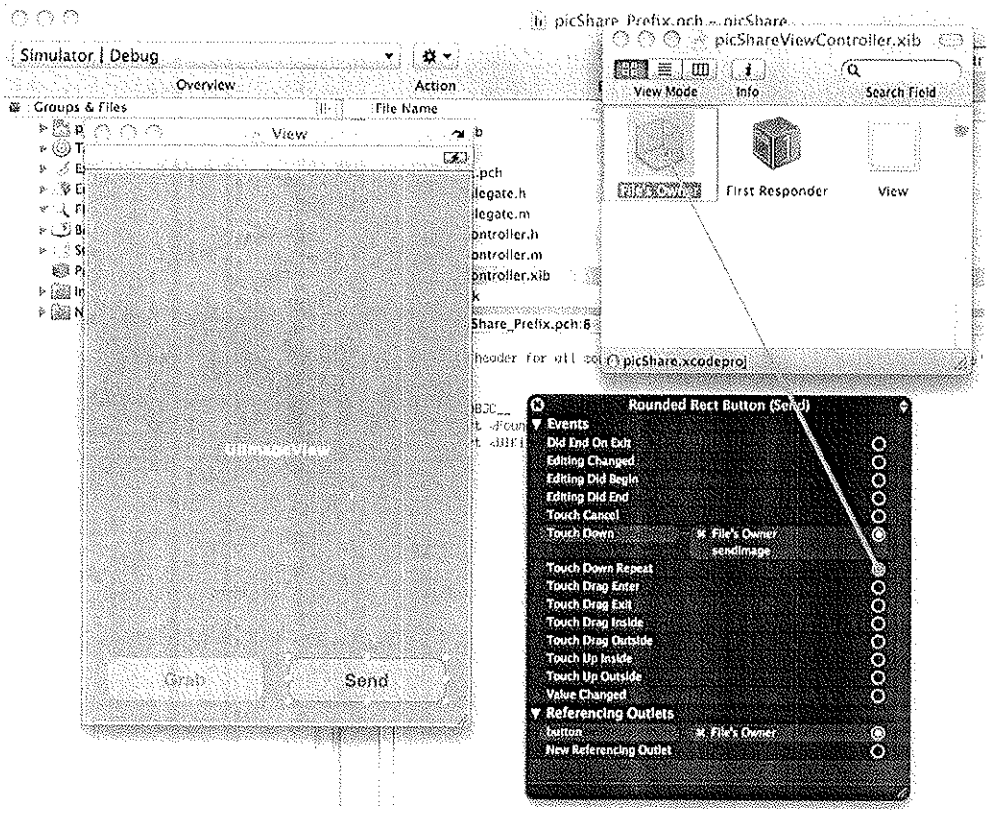

Figure 9 Xcode interface builder

After familiarizing ourselves with Xcode we continued to our next task i.e. building an application that displays the values of accelerometer's x, y and z coordinates. After running the application the device showed:

X: -0.181122
Y: -0.796936
Z: -0.470917

X:
Y:
Z:

Figure 10 First Dummy Application Results

After completing this we developed the main skeleton for our project .Our initial approach is explained in the following diagram.

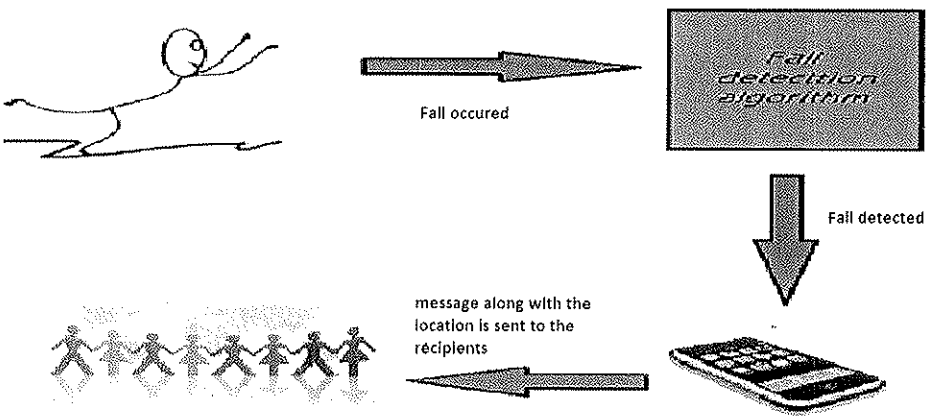

Figure 11 Initial Approach

(12) Apple doesn't allow automated (or even partially automated) SMS and dialing operation for security reasons therefore the above method was not feasible. The only other approach we had was to set up a server on the internet that sends messages using a SMS gateway service.

Our current approach could be explained in the following diagram:

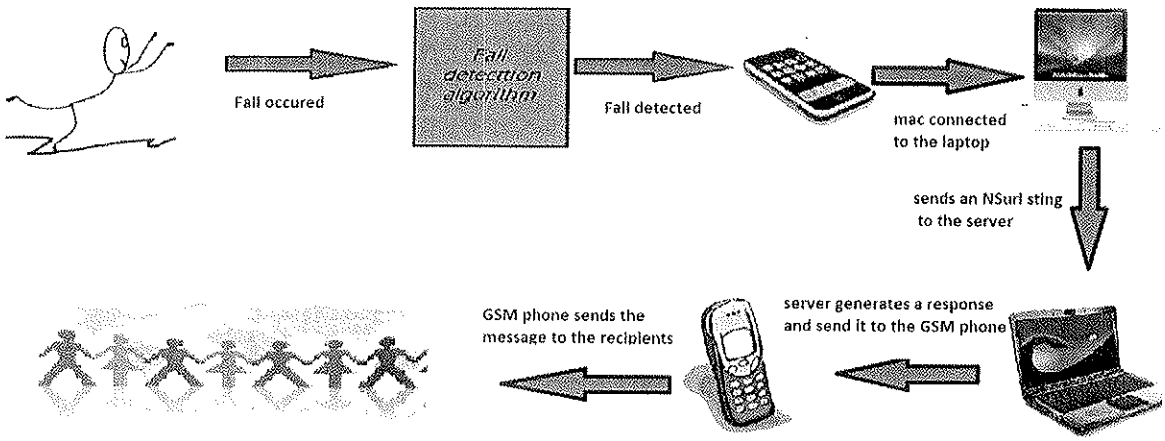

Figure 12 Final Approach

The basic steps involved in the implementation of our methodology are:

1: Intensive experimentation

2: Threshold determination

3: Setting up the server and SMS gateway

4: Location

5: GUI design and user setting

3.1 Intensive Experimentation

Our first step was to save the readings from accelerometer in a file to generate graphs for different postures. Though it was easy to write the contents to a file but reading that file from the device was tricky. After a week's research we found the solution and were able to read the file from the device and save it on the computer.

We had all the required ingredients so we started experimentation. We took the readings by placing the iPhone upright in the chest pocket. For accuracy we changed the code to have 20 values per second. The readings were mainly taken for

sitting and falling positions. Initially we were only taking x y and z coordinates values but the results were not as expected so we calculated the total acceleration in the code using the formula:

$$|AT| = \sqrt{|AX|^2 + |AY|^2 + |AZ|^2}$$

We did the experiments again and saved the total acceleration values in the file along with x , y and z coordinates values. In the end we had 10 graphs for each position.

3.2 Threshold Determination:

Once we had the graphs we compared all of them to get the range of values as our initial guess for the threshold value. We selected the most suitable value by checking these values for FN, TP, FP and TN through experimentation.

Table 1 FN, FP, TP, TN values for different threshold values

Threshold	FP	TP	FN	TN
0.65	2	4	2	3
0.66	1	3	3	5
0.67	2	6	1	3
0.68	4	3	2	4

**FN: false negative (fall has occurred but the device does not detect it)*
**FP: false positive (fall has not occurred but the device detects it)*
**TN: true negative (fall has not occurred and the device does not detect it)*
**TP: true positive (fall has occurred and the device detects it)*

In general, the lower the both FN and FP are, the better the performance is. Therefore by comparing the FN and FP of the above 4 values we came to the conclusion that 0.67 is the best option for threshold value as it gives the best balance between the FN and FP.

3.3 Setting up the Server and the SMS gateway

After dealing with the selection of threshold value our next task was to set up a server for sending automated SMS. For this we used Ozeki NG-SMS (13) gateway service that uses a GSM phone connected to the laptop and an inbuilt server to send messages. To make this work we needed to configure Internet Information Services (IIS) and install .NET Framework 3.5 and Ozeki NG-SMS gateway on our laptop. After doing this we connected our GSM phone with the laptop and started the Ozeki NG-SMS service.

Our next step involved connecting the Mac (where our main iPhone application resides) and the laptop for that we needed a static IP which can be called from our application to the machine where the Ozeki NG-SMS gateway resides. We tried a lot to get the static IP but couldn't get it because of SEECS policy instead we connected our laptop and Mac through the Ethernet.

Once the connection was established between the laptop and the Mac an HTTP request was send to the server using an NSurl string in our code. In accordance with the request the server generated a proper response for the SMS gateway which then sends a message to the recipient mentioned in the URL string using the GSM phone attached to the server. The message body contains the content of "messageData" field of URL string.

```
NSString *url = [NSString
stringWithFormat:@"http://10.3.70.238:9501/httpapi?action=sendmessage&u
sername=admin&password=abc123&messageType=SMS:TEXT&recipient=0
3334562035&messageData=Fall is detected";

NSURLRequest *request = [NSURLRequest requestWithURL:[NSURL
URLWithString:url]];
```

3.4 Location

Another important element of our project was to get the current location of the user and send it to the emergency contact along with the message. To achieve this purpose we added a CoreLocation framework (14) to our application. There are few classes in the Core Location framework and CLLocationManager is the most important one. It handles sending out updates to a delegate anytime the location is changed.

```
- (void)locationManager:(CLLocationManager *)manager
   didUpdateToLocation:(CLLocation *)newLocation
        fromLocation:(CLLocation *)oldLocation
```

Whenever a new location is available the above method is called and we will get both the old and new location in the CLLocation object. These are displayed in the label in the mainviewcontroller using the following code:

```
locLabel.text = [NSString stringWithFormat:@" %f %f"
,newLocation.coordinate.latitude ,newLocation.coordinate.longitude];
```

3.5 GUI Design and User Setting

In our last step we finalized a user interface for our application .There are two views in our application; mainviewcontroller and a secondview. The mainviewcontroller contains four buttons (start, settings, ufall and stop) , four labels (total acceleration , location and two empty labels). When the application starts mainviewcontroller is loaded displaying the current location of the user. The user can switch to the secondview by pressing the button "settings".

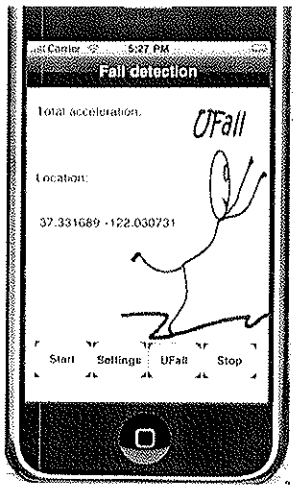

Figure 13 main view

The secondview contains a label (threshold) , a textfield (for threshold value) and a button (back).This view allows the user to set a threshold value and switch back to the first view.

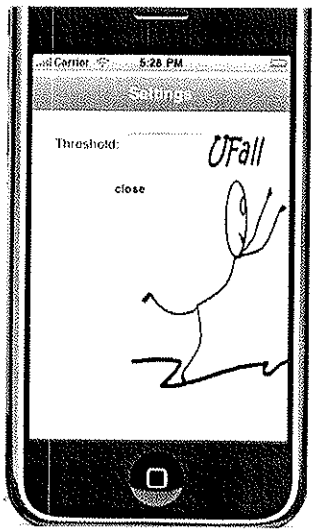

Figure 14 second view

Once the threshold is set the start and stop button of the first view are used: the "start" button starts comparing the acceleration value with the threshold value until the acceleration value is greater than the threshold value. Once the acceleration value exceeds the threshold the application shows an alert displaying fall is detected and the application stops automatically. The "stop" button stops the application.The uFall button works in the simulator to send message to the emergency contact.

4. Results

After integrating all individual software components, a lot of experimentation was required to find an accurate approximation for the threshold value.

The iPhone was placed upright in the front pocket of the test subject. This was to set a standard for getting sensitive and responsive data. For the experiments we selected a cushioned surface for the safety of the subject .Later, we adjusted the value for a harder surface.

After setting up everything, the application was started by pressing the "start" button and the test subject was made to fall or sit on a cushioned surface. The application was stopped such that the duration between start and stop was kept constant i.e. 10 seconds .A low pass filter was applied on all the accelerometer values to negate the effect of gravity .The code was adjusted to give 20 values per second and these were written in a file saved on the device. After every experiment the device was connected to the Mac to get the values for the graphs. The process was repeated five to ten times for every posture. Initially when we were taking the forces in x, y, z directions we got the following results :

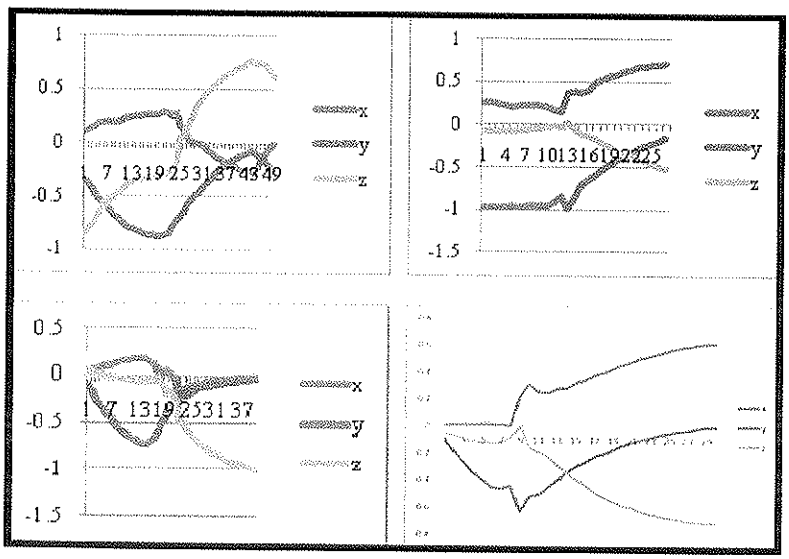

Figure 15 Falling

The results of these graphs did not meet our expectations due to inadequate use of the low pass filter . As the person can move in any direction at any given movement it was impossible to make a conjecture on an appropriate value for the threshold by analysing accelerometer values in the three coordinates. To overcome this problem we altered our code to calculated the total acceleration on the device using the accelerometer values and did the experimentation again. The whole process was repeated but this time the graphs were for the total acceleration as shown below:

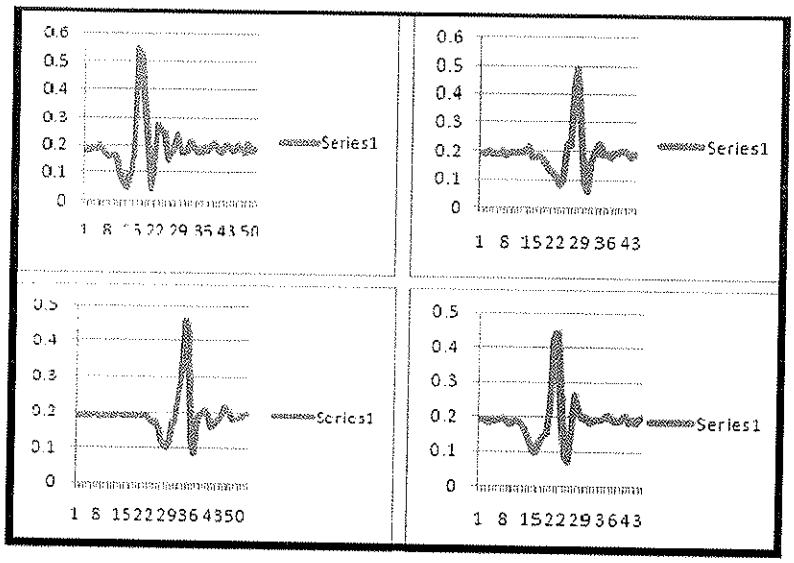

Figure 16 Sitting

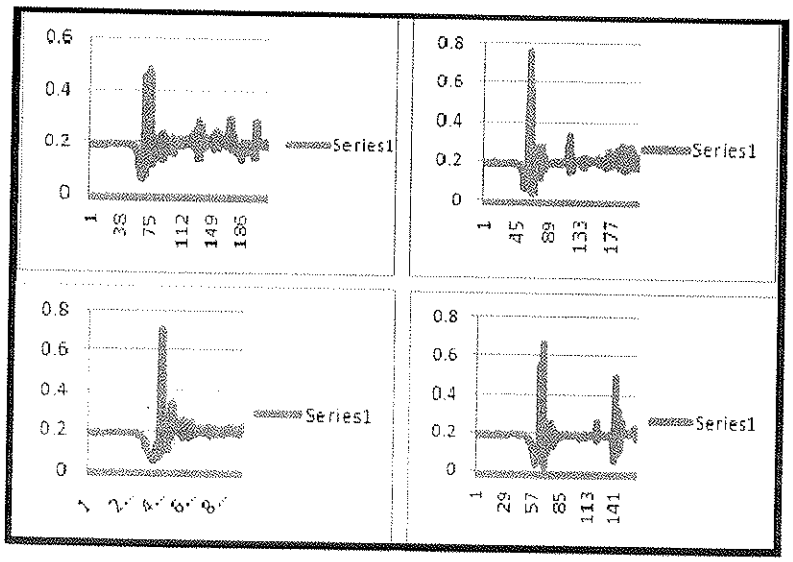

Figure 17 Falling

The experiments were done for two postures and the values were compared to determine the range of possible values for the threshold .According to the graphs during a fall the maximum acceleration achieved is in the range 0.65 – 0.68.

Figure 18 Fall experiment setup

After having the range of possible threshold values we had to choose the most suitable value for the threshold. For this purpose we added a fall detection alert and a textfield for threshold in our application.

We did the experiments using each value from the range 0.65 – 0.68 as a threshold value to check which value detects fall more accurately. The experiment was done ten times for each value. Each time the test subject entered the threshold value and pressed the start button the threshold values was compared with the total acceleration of the device. Once the application was running the test subject tried different postures to see different possibilities:

- How many times fall has occurred but the device didn't detect it.(False negative)
- How many times fall hasn't occurred but the device has detected it.(False positive)
- How many times fall has not occurred and the device did not detect it.(True negative)
- How many times fall has occurred and the device detects it.(True positive)

The table below shows the results of these experiments:

Table 2 FN, FP, TP, TN values for different threshold values

Threshold	FP	TP	FN	TN
0.65	2	4	2	3
0.66	1	3	3	5
0.67	2	6	1	3
0.68	4	3	2	4

By using the concept that the best value is the one that gives lower FN and FP it was deduced from the table that the most suitable threshold value was 0.67.

5.Discussion

Overall the fall detection application is working up to our expectations. Both the server part and the application is performing according to the specifications.

Initially we just tried to retrieve the value of force acting in X, Y and Z directions. We also applied low pass filter on these values to eliminate the effect of gravity and then made the test subject to fall on a mattress or a sofa. After each fall we connect the iPhone with mac, retrieve all the values and plot graphs on each fall. We did this for 4-5 times. After getting the graphs we tried to determine a threshold value on the basis of these graphs but we were unsuccessful. As a lot of time was wasted in this so we started thinking on alternative approaches. After some extensive brainstorming we came to a solution i.e. to take the total acceleration for determining the threshold. We calculated the total acceleration through the formula:

$$|AT| = \sqrt{|AX|^2 + |AY|^2 + |AZ|^2}$$

After doing this we repeated the whole procedure i.e. experiments in the Siting sharply and falling positions with the total acceleration values and came to a conclusion that 0.67 is the best option for the threshold value as it gives the minimum false negatives (fall has occurred but the device dose not detect it) and false positives (fall has not occurred but the device detects it).

Another problem that we faced was the inbuilt inability of iPhone OS which prevents a message to be sent without a tap. This was a serious problem as it was a major requirement of our application to send a message when the application detects a fall and without the user's consent. To solve this issue we came up with a idea of using a server which can send message on behave of the iPhone. We then setup a whole server on a laptop using Ozeki NG-SMS gateway service , which uses a GSM phone to send messages on behave of the iPhone. Now another challenge that we faced was in getting a static IP which can be used to call the service from the mac. As it was against SEECS policy to grant a static IP to some

student so we had to find another alternative. The most suitable option left was to connect mac and the laptop through the Ethernet. It did work fine but it has only one issue i.e. a message can only be sent using the simulator and not the iPhone as the iPhone uses WiFi and the mac and the laptop were connected through the Ethernet.

6. Conclusion

Our final application in spite of all the hurdles worked as predicted and it met our expectations. It correctly detects a fall on the basis of the given threshold value and displays an alert of fall detected. It is user friendly as the user can give the threshold value of their choice too. The application has a start and stop button to start and end the application. The application is by no means perfect and has some drawbacks but it is working as per our anticipations. Not only has it been a great learning experience, it has helped us appreciate those who spend months developing useful applications for the benefit of mankind.

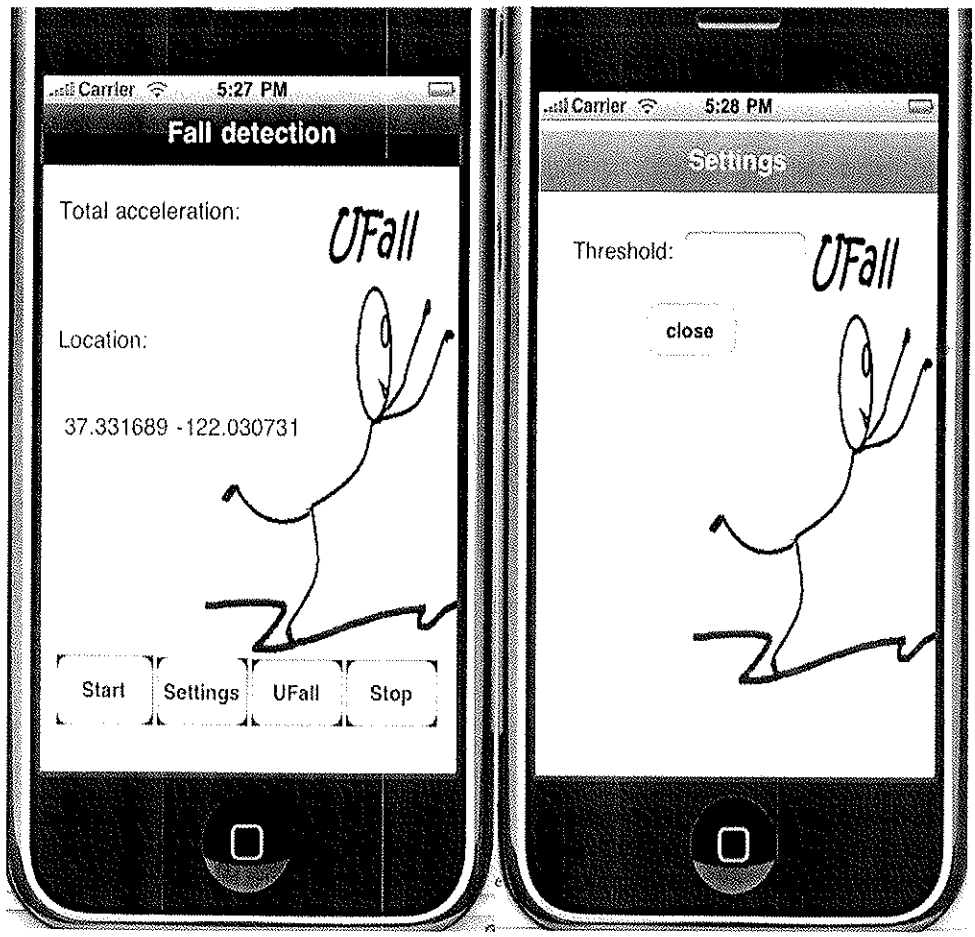

Figure 19 Final Product

7. Recommendations

7.1 Getting a Static IP

Currently we have used Ethernet to establish connection between a computer (connected to the server on the internet) and a Mac (where our application is saved).This set up does not allow us to send automated SMS from the device instead the simulator sends the SMS. However, with the availability of a static IP it is possible to send SMS to the emergency contacts using the device which would be an ideal case for this application.

7.2 Using Reverse Geocode

Our application gives the current location in terms of longitude and latitude. The location would be easier for the reader to comprehend if it is given as a geographical location instead of just longitude and latitude. This can be achieved by using the reverse geocoding technique.

(15) Reverse geocoding is a method that involves reverse coding of a point location to a comprehensible address thus making the recognition of nearby street addresses, places, or areal subdivisions possible.

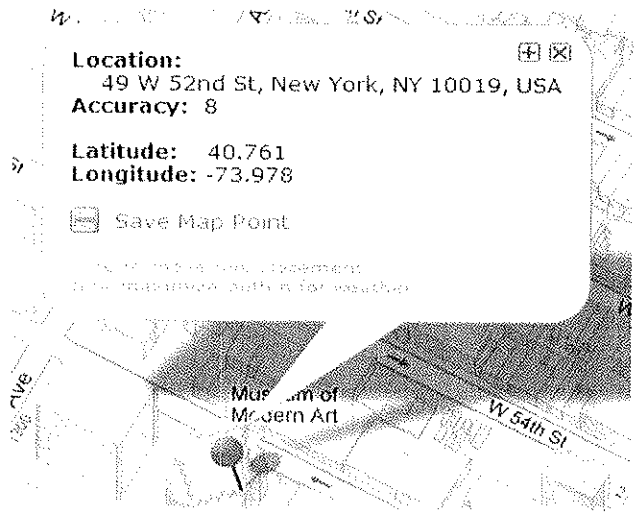

Figure 20 Reverse geocoding

7.3 Using Pattern Recognition Techniques

For determining a more efficient threshold the pattern recognition techniques can be used. This will result in more accurate and precise fall detection by analyzing the entire fall pattern from the graph instead of a single peak value.

7.4 Improving the interface

Interface is a place where human interacts with the machine and therefore must be built such that it fulfills all the needs of the user. (16) User interface is what makes a product accepted or rejected in the market. An attractive, easy to use and interactive user interface is what helps in winning the user acceptance.

More features can be added that will allow the application to work differently in a given scenario for e.g. user profile (1). Another view can be added to the application that will get all the data like name, age and contacts from the user to make a user profile. This will make the application choose a threshold according to the user's age.

Another useful feature would be to add radio buttons that will allow the user to set the sensitivity level of the system himself depending on how active will he/she be on that day.

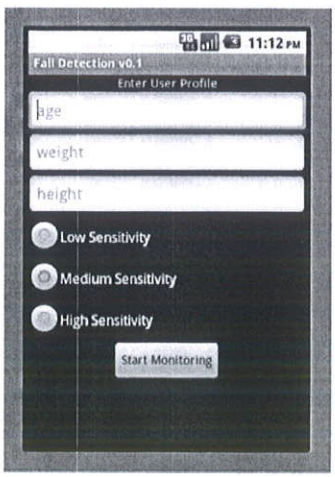

Figure 21 Enhanced interface

7.5 Adding Multiple Contacts

Our current application does not allow the user to have multiple emergency contacts. However, the application can be expanded by allowing the user to add and delete multiple emergency contacts.

7.6 Customized Message Content

The application can be adjusted such that the user can save the text of his choice to be sent as message content in the SMS.

References

1. **Gonzales, Ian James Daniel.** *Fall Detection Using a Smartphone.* 2010.

2. Fallwatch project. *Fallwatch project.* [Online] http://www.fallwatch-project.eu/.

3. Smartphone. *wikipedia.* [Online] http://en.wikipedia.org/wiki/Smartphone.

4. medgadget. [Online] http://medgadget.com/archives/2010/04/ifall_an_android_fall_detection_app_1.html.

5. **Jia, Ning.** *Detecting Human Falls with a 3-Axis Digital Accelerometer.* 2009.

6. **Qiang Li, John A. Stankovic, Mark Hanson, Adam Barth, John Lach, Gang Zhou.** *Accurate, Fast Fall Detection Using Gyroscopes and Accelerometer-Derived Posture Information.* 2009.

7. **Jiangpeng Dai, Xiaole Bai , Zhimin Yang , Zhaohui Shen and Dong Xuan.** *PerFallD: A Pervasive Fall Detection System Using Mobile Phones.*

8. **Tong Zhang, Jue Wang, Ping Liu and Jing Hou.** *Fall Detection by Embedding an Accelerometer in Cellphone and Using KFD Algorithm.* 2006.

9. Reverse geocoding. *Wikipedia.* [Online] http://en.wikipedia.org/wiki/Reverse_geocoding.

10. Reverse Geocoding. *Wikipedia.* [Online] http://en.wikipedia.org/wiki/Reverse_geocoding.

11. User Interface. *Wikipedia.* [Online] http://en.wikipedia.org/wiki/User_interface.

12. how-to-programmatically-send-SMS-on-the-iPhone. *stackoverflow.com.* [Online] http://stackoverflow.com/questions/10848/how-to-programmatically-send-SMS-on-the-iPhone.

13. SMS gateway service. [Online] http://www.ozekiSMS.com/index.php?owpn=230.

14. Location in iPhone. [Online] http://www.switchonthecode.com/tutorials/getting-your-location-in-an-iPhone-application.

15. Objective C. *roseindia.* [Online] http://www.roseindia.net/iPhone/objectivec/objective-c-windows.shtml.

16. Compiling objective C. *roseindia.* [Online] http://www.roseindia.net/iPhone/objectivec/compiling-objective-c.shtml.

17. What's new. *developer.apple.com/.* [Online] http://developer.apple.com/technologies/tools/whats-new.html.

18. Xcode projects. *developer.apple.com.* [Online] http://developer.apple.com/tools/xcode/xcodeprojects.html