An Accelerometer Based Fall Detector: Development, Experimentation, and Analysis

Garrett Brown
Computer Science
University of Michigan
garretto213@hotmail.com

Faculty Mentor: **Ruzena Bajcsy**
Research Supervisor: **Mikael Eklund**
Summer Undergraduate Program in Engineering at Berkeley (SUPERB) 2005

***Abstract*— The financial exertion and physical requirements required to provide the current level of health care to the rapidly growing elderly citizen population will create tremendous strain on the current health care systems, thus various ideas have been proposed to provide the appropriate level of care in a more efficient manner by taking advantage of current technologies. To facilitate the safety, security, and continual supervision of a constant care environment while still allowing the user to retain their ability to live at home, the ITALH, Information Technology for Assisted Living at Home, project has been initiated. This paper describes a fall-detection sensor that will be implemented into a larger sensor network called SensorNet, which is being developed to safely and accurately monitor the user while still allowing complete privacy and security. Falling is one of the most significant causes of injury for elderly citizens, and one of the reasons why many otherwise healthy individuals are forced to leave the comfort and privacy of their own home to live in an assisted-care environment. By utilizing acceleration values corresponding to the user's body motion, much effort has been put towards developing a robust algorithm to accurately detect a fall by the user.**

I. Introduction

A. Motivation

There is an impending influx of elderly citizens due to the maturation of the baby boomer generation. There will soon be a much larger ratio of citizens over 65 to all citizens in North America than ever before (see Fig. 1).

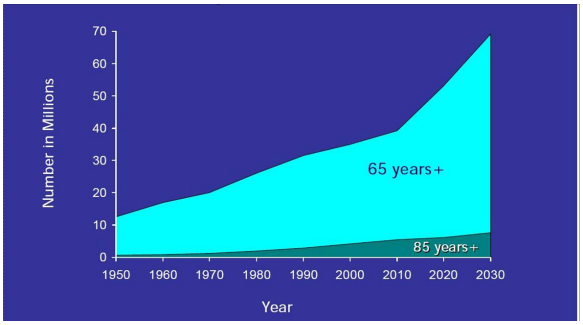


Fig. 1. Health, United States, 1999, U.S. Bureau of the Census plot showing the projected increase of the percentage of elderly citizens in the U.S. population

This creates a necessity to begin development of more efficient and cost-effective methods of caring for them adequately. The financial exertion and physical requirements required to provide the current level of care to such a large population are far too great to be feasible, thus various ideas have been produced to provide the appropriate level of care in a more efficient manner by taking advantage of current technologies. To help facilitate the safety, security, and continual supervision of a constant care environment while still allowing the user to retain their ability to live at home, the ITALH project has been initiated. The Information Technology for Assisted Living at Home (ITALH) project is aimed at increasing quality of life with a focus on the home through better support for elderly citizens who want to stay in their own homes without forgetting the support of emerging mobile lifestyle. This is made possible by utilizing basic design for all concepts and providing general solutions for the usability and applications of new technology for home and health [2], [3]. This ITALH initiative has been an umbrella project that includes the IVY project which is concerned with looking at fall sensors and SensorNet which is concerned with developing an integrated, wireless sensor to accurately monitor the user while still allowing complete privacy and security (see Fig. 2). The focus, thus far, from the IVY and SensorNet projects have been geared towards developing a fall-detection device [1].

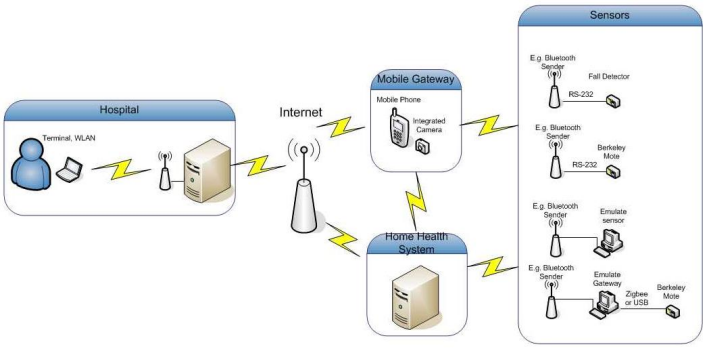


Fig. 2. Physical representation of the SensorNet concept

B. Why the fall sensor?

Falls are the leading cause of fatal and nonfatal injuries to older people in the U.S. Each year, more than eleven million people over sixty-five fall - one of every three senior citizens [4]. Treatment of the injuries and complications associated with these falls costs the U.S. over twenty billion annually [4]. Due to the fact that sixty percent of falls occur at home and the danger and severity of falling and the possibilty of not having any assistance in case of unconciousness or extreme injury are primary reasons why many otherwise healthy individuals are forced to leave the comfort and privacy of their own home to live in an assisted-care environment (Forty percent of nursing home admissions are due to falls [4]). Furthermore, a fall can cause psychological damage even if the senior is not physically injured. Fall researchers describe a fear of falling cycle in which after a fall seniors become so afraid of falling again they limit their activities. This in turn decreases their fitness, mobility and balance and leads to decreased social interactions, reduced satisfaction with life and increased depression. This fear cycle then increases the risk of another fall [8]. A fall-detection sensor device has been created and modified to now reside on the user's hip in the form of a waist belt. This device provides continuous and instantaneous data corresponding to the changes in a three-dimensional acceleration matrix surrounding the user's body. There has been much analysis of the how to use the acceleration change values to develop a robust algorithm to accurately and consistently detect falls by the user. This paper describes some of the methods used and the results achieved.

II. BACKGROUND

A. Cartesian Coordinates

Cartesian coordinates are rectilinear two-dimensional or three-dimensional coordinates which are also called rectangular coordinates. The three axes of three-dimensional Cartesian coordinates, conventionally denoted the *x*-, *y*-, and *z*-axes are chosen to be linear and mutually perpendicular(see Fig. 3). In three dimensions, the coordinates *x*, *y*, and *z* may lie anywhere in the interval (-∞, ∞) [5].

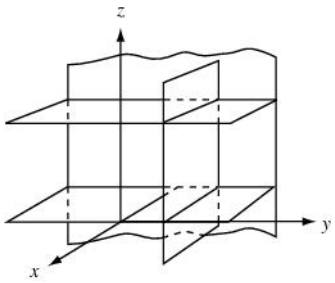


Fig. 3. Cartesian Coordinate Plane

B. Spherical Coordinates

Spherical coordinates are a system of coordinates that are natural for describing positions on a sphere or spheroid (see Fig. 4). Define θ to be the horizontal angle in the *xy*-plane from the *x*-axis with $0 \leq \theta \leq 2\pi$ (denoted λ when referred to as the longitude), ϕ to be the vertical angle from the z-axis with $0 \leq \phi \leq \pi$, and *r* to be distance (radius) from a point to the origin [6].

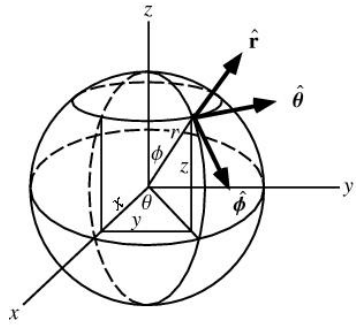


Fig. 4. Spherical Plane

$$r = \sqrt{x^2 + y^2 + z^2}$$

$$\theta = \arctan(y/x)$$

$$\phi = \arcsin(\sqrt{x^2 + y^2}/r) = arccos(z/r)$$

In terms of Cartesian coordinates,

$$x = r\cos(\theta)\sin(\phi)$$

$$y = r\sin(\theta)\sin(\phi)$$

$$x = r\cos(\phi)$$

C. Dot product

The dot product can be defined for two vectors *X* and *Y* by

$$X \cdot Y = |X||Y|\cos(\theta)$$

where θ is the angle between the vectors and $|x|$ is the norm. It follows immediately that $X \cdot Y = 0$ if *X* is perpendicular to *Y*. The dot product therefore has the geometric interpretation as the length of the projection of *X* onto the unit vector *Y* when the two vectors are placed so that their tails coincide [7].

III. EQUIPMENT

A. Sensor Board

1) Specifications: The idea for a device to register continuous acceleration values originally started through an earlier study in the form of a wrist device. Due to the continuous movement of the arm and the unpredictability of a user's motor response in the case of a fall, however, it became readily apparent that a wrist sensor would not be stable enough to provide accurate fall detection capabilities. The next prototype for the device then became a rectangular sensor board. After some testing with various positions such as the chest, the neck, and the waist, it was confirmed that the waist would be the most stable position to monitor movement [9].

The fall sensor devices are now in their second generation of development. The first generation device was developed under the IVY project and used MicaDot technology with wireless communication. The second generation, the “GPSADXL” is a microcontroller-based data logging system that integrates two +/- 10g MEMS ADXL210 accelerometers with a compact GPS module and four megabytes of static RAM. The device is powered by three AAA 1.2 volt rechargable NiMH batteries and measures 2.50” x 3.75” x 0.70” (see Fig. 5). The battery life of the sensor board if it was set to continuously stream data is around 10 hours with approximately 4 hours of data recording/storage with a decrease in sensitivity to acceleration of only about one-twentieth of the Gravity vector, or 1/20 *G*, during its battery life. The MEMS accelerometers are arranged at a ninety degree angle to each other to allow the measurement of acceleration in three dimensions and provide acceleration data at a rate of 80 sample groups per second. The GPS module supplies time of day and latitude and longitude information, as well as ”lock” status and the number of satelites in view. If the data to be transmitted is accelerometer data then it is converted to ASCII hex form and is tranmitted as serial RS-232 data to an external device. If the data is GPS or a text message, then it is already in ASCII form and is transmitted as is. The serial port settings are: 115,200 baud, 8 data bits, no parity, no flow control. The fall sensor device is able to connect into the SensorNet web by using BlueGiga Bluetooth RS-232 cable replacement devices to connect to a control device such as a laptop or a mobile phone.

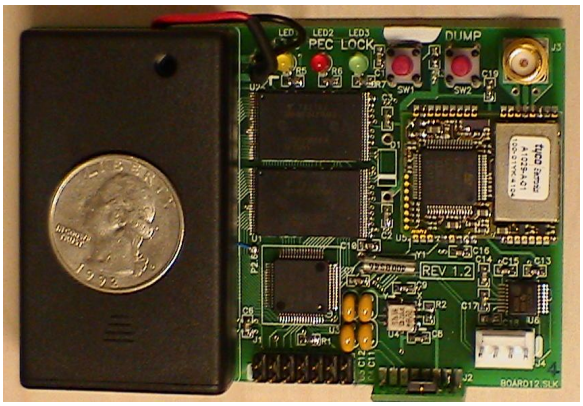


Fig. 5. Fall Sensor

2) Calibration: Before any data analysis for an algorithm can be carried out, it is crucial that the data be meaningful and correctly interpreted. In order to do this, it is necessary to calibrate the device to specify what values are being worked with. The algorithm for calibrating the board to get the initial starting values proceeds as follows:

1) Hold the board in an upright position for five seconds(the *x* and *y* values should be zero) (see Fig. 6)
2) Average the *z* acceleration data over this period to get the upright *z* values
3) Average the *x* and *y* acceleration data to get the initial zero *x* and *y* values
4) Flip the board over completely into the inverted position and hold for five seconds(the *x* and *y* values should again be zero)
5) Average the *z* acceleration data over this period to get the downward values
6) Take the difference between the upright and downward *z* values and divide by two to get the 1 *G* gravity vector
7) Take the average of the upright and downward *z* values to get the initial zero *z* value

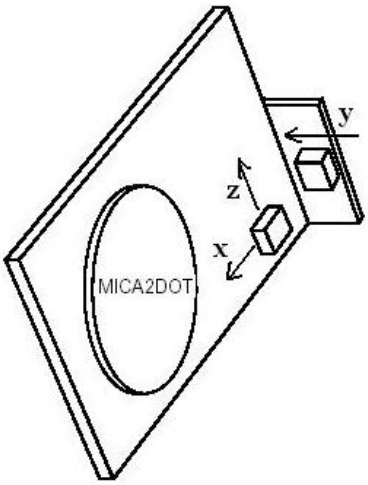


Fig. 6. The fall sensor board in an upright/inverted position

The procedure for doing this involves simply holding the board in an upright and then inverted position for a short amount of time as prompted by blinking lights on the board. This allows the value of the 1 *G* gravity vector to be determined and provides for the initial zero starting values for the *x*, *y*, and *z* axes to be initialized.

3) GPS Capabilities: The sensorboard has a built in GPS chip which can provide location data. The board originally provided data only when it was in dump mode, meaning it stored all it’s data into the SRAM chip on the board and was only accessable after a memory dump. I was able to reprogram the board’s functionality to produce GPS data in streaming mode as well. This means that, in the case of a fall, the board can emit a GPS tracking signal specifying the user’s location, the time of day, and information regarding how accurate the GPS location data is. Since the GPS capability only works outside of the home due to the fact that it needs to be in view of a satellite, this feature can be utilized when the user leaves the house. Tracking inside the house may be done in the future by utilizing Berkeley Mote technology [10] or cameras either on the user’s mobile phone or positioned throughout the house.

4) Output Specification: When the complete sensor network is in place, it will be crucial to have specifications for any type of communication interaction between the devices and the central processing device. Without this precise communication protocol, critical information could potentially be misread and/or misinterpreted which could have fatal results. The fall sensor has various types of data it is capable of producing as of now, therefore it is already necessary to take these actions. To aid in this process, we have implemented a communication protocol onto the fall sensor. Thus, when the fall sensor is connected up to the monitoring parent device, the parent

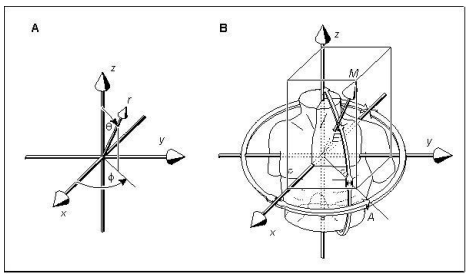
device can be designated to interact with the specified format of the output from the fall sensor. This ensures that when the fall sensor sends information out, the monitoring device will know whether the information is valid or invalid, and in the case of verified validity, will know what is being expressed and then take the appropriate action.

5) Bluetooth Interaction: Thomas Hansen has created a Bluetooth Server which searches for incoming signals from any type of connection seeking sensor device. The fall sensor-board has been programmed with the ASCII interface specifications to initiate the bluetooth device into command mode and from there to command the bluetooth device to send out a seek signal to try and connect to a parent device. Therefore, in the case of a fall, the fall sensor has the inherent capability to seek out and initiate a connection. Once connected, the fall sensor can provide any information required such as GPS data, fall detection alert, or the commands to carry out an action designated by the user.

IV. METHODOLOGY

A. Data Analysis

From the streaming data continually provided by the fall sensor, the challenge was how to use it with the most effectiveness. In a regular, standing position, the only value noticeable is the effect of the gravity vector *G* on the *z*-axis. In theory, if the user is wearing the device correctly, the *x*-axis and *y*-axis will be zero. Once this initial position has been provided by the calibration procedure, it is then possible to use the dot product of the *z* vector and the *G* vector to determine how the user is oriented in comparison to the upright initial position. Furthermore, it is also possible to convert the acceleration data values into spherical coordinates and therefore have a general picture of the user's instantaneous orientation with respect to the real world perspective.



Another topic of debate with respect to the data analysis was that if it was better to look at each of the *x*, *y*, and *z* axes separately when monitoring for a large acceleration or to simply look at the magnitude vector $r = \sqrt{x^2 + y^2 + z^2}$ of all three axis together. There is a concern on whether in some cases of unusual acceleration the magnitude vector would be larger than it should be and thereby have adverse effect on the magnitude detection algorithms. To analyze and provide comparison of these two methods the Simple Magnitude algorithm used the individual axis monitoring approach while the Advanced Magnitude algorithm used the magnitude monitoring approach.

B. Algorithm Development

1) Challenges: When creating an algorithm to carry out such a critical task as trying to detect falls, there are always very important issues to keep in mind. The first major concern was to avoid false positives. False positives exist when a test reports, incorrectly, that it has found a signal where none exists in reality. The effects that repeated false positives may have are unnecessary notification of emergency personnel causing embarrassment and frustration of the user. If the user is annoyed to the point of not wanting to wear the fall sensor, it will be useless in predicting falls. The second major concern was trying to avoid false negatives. False negatives are the direct opposite of false positives, meaning that they exist when a test does not report that it has found a signal where one does indeed. Obviously, a fall detection sensor that does not detect falls is utterly useless. Other concerns with the algorithm are that it must be robust, customizable for individual needs, and highly accurate and reliable.

2) Conjoined Angle Change and Magnitude Detection: Our first algorithm consisted of observing changes in angles in conjunction with a crossing of a set acceleration threshold within the same time interval. The angle change was the flag which initiated the algorithm observation, and when conjoined by a large acceleration within the same time frame was then labeled as a fall. The reasoning behind this initial approach was that in most fall scenarios the user would have some type of angle change created by falling to be inlined with the ground from an initial standing and sitting position. We also thought this would be good since in the case of running, jumping, stepping up on a stool, or other such actions, although there would be a noticeable aceleration change, the hip positioning angle would stay relatively uniform and thus not set of a fall alarm (see Fig. 7) . The functionality behind this method was to observe a significant change in the user's orientation angles, look for a large acceleration within the same time inteval, and when both are present, classify as a fall. The basic algorithm design for carrying this out proceeds as follows:

1) Look for a *Significant Angle Change* within a designated *Time Interval Length*
2) Once a *Significant Angle Change* is encountered, look for a breech of a *Large Acceleration Threshold* within the same *Time Interval Length*
3) If both of these actions occur within this *Time Interval Length*, classify as a fall and take designated actions

The parameters to be tested in this algorithm were *Significant Angle Change*, *Time Interval Length*, and *Large Acceleration Threshold*.

Upon testing this algorithm with real life simulations, we realized that we must take into account that changes in angles and acceleration can be part of everyday activities such as picking something up off the floor, stretching, or sitting down and therefore must not be detected as a fall. Furthermore, the act of getting up off the floor after a fall would also involve a large angle change coupled with a large acceleration and thus be considered as a fall, totally opposite of what we wanted. Thus further algorithm refinement was necessary.

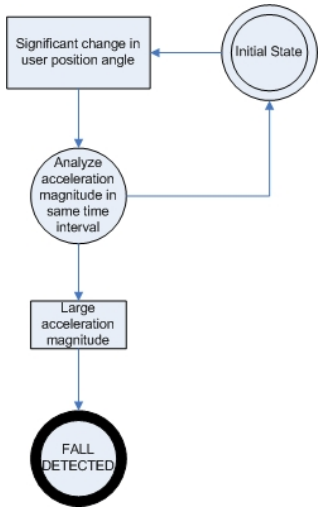


Fig. 7. Conjoined Angle Change and Magnitude Algorithm

Although we felt that a fall-detection algorithm could be developed in a simpler fashion by starting with a large magnitude of acceleration and then doing further orientation analysis, we still felt the idea of angle changes would be very useful given a meaningful initial calibration. Given a real world perspective of the user, meaning providing transformations to ensure that the user is always being considered with respect to an erect straight-standing posture position, the accelration data in terms of spherical coordinates could be very useful. For instance, the angle Φ can determine just how far the user may be leaning over and if the user is laying on the floor. If an accurate real world model were presented at the beginning of the testing and if the acceleration data provided were interpreted correctly in terms of orientation changes with a real world perspective, it would be possible to provide a continuous real world model of the user which could accurately follow precise changes in position. This capability would undoubtedly be useful in trying to determine whether the user has sustained a fall or is moving normally. The future of this ability extends also beyond fall-detection and into a real world movement analysis realm where in the future, real time analysis can provide discreet information on wether the user is walking, running, etc.

3) Simple Magnitude Detection: The first idea for a acceleration magnitude threshold fall-detection algorithm was to detect a large change in acceleration from the starting orientation, and then provide subsequent analysis on the user's orientation to confirm whether or not the change in acceleration had indeed been a fall or may have been some other action. Since almost all falls result in the user lying on the ground in some way, it seemed that we could easily use this fact to catch these falls (see Fig. 8). The functionality behind this method was to observe a large change in the acceleration, wait until the large acceleration ended, analyze orientation, and if the orientation revealed the user to be horizontal with the ground, classify as a fall. The basic algorithm design for carrying this out proceeds as follows:

1) Look for a breech of a *Large Acceleration Threshold*
2) Wait until the large acceleration dissipates and we get relatively *Normal Acceleration*
3) Provide a short time interval *(around 12 seconds)* for the user to get acclimated
4) Analyze the user's orientation
5) If it is determined that the the user has *Orientation Horizontal with the Ground*, classify as a fall

The parameters to test in this case were *Large Accleration Threshold*, *Normal Acceleration*, and *Orientation Horizontal with the Ground*.

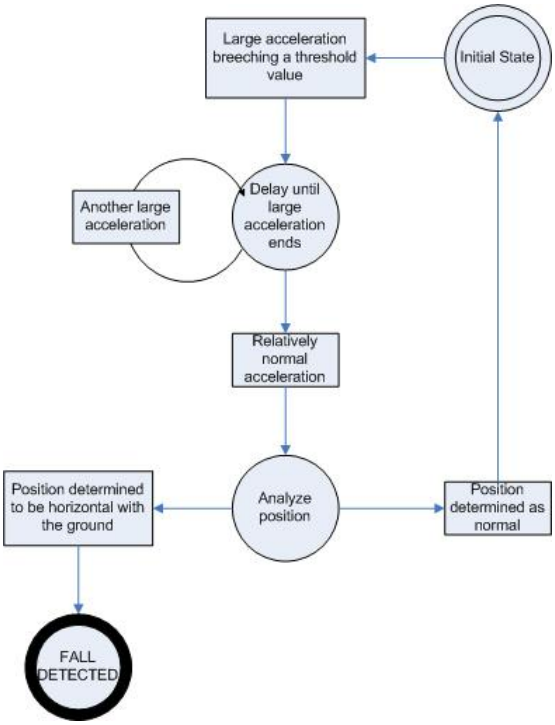


Fig. 8. Simple Magnitude Algorithm

Based on this method, almost all falls with a noticeable change in user acceleration magnitude in which the user ended up lying horizontal with the ground were correctly predicted. The weaknesses discovered were that no fall would be predicted if the user fell in such a way that he was not parallel with the ground. This is very important as in various cases during a fall, a user will try to grasp a wall, bed, or other object and end up slumping next to the object rather than flat on the floor. Therefore, it is crucial to have another analysis of orientation to try and catch these cases as well.

4) Advanced Magnitude Detection: From the weaknesses we discovered with the Simple Magnitude Detection algorithm in detecting a fall in which the user did not end up horizontal with the floor, we changed the algorithm to compensate for this. The initial magnitude detection and position analysis was the same, but we added an extra measure of position analysis for the special cases. In any situation after a large magnitude of acceleration in which the position analysis revealed the user deviating from a completely upright position, we would continue with further position analysis. If, in the case of a deviation from uprightness, the user remained relatively inactive for a short while, meaning no significant acceleration,

then we would analyze the orientation again. If the orientation still differed from uprightness then we asumed the user may have slumped against a wall, fallen on a flight of stairs, or fallen onto some type of object and therefore predicted this as a fall (see Fig. 9). The functionality behind this method was to observe a large change in the acceleration, wait until the large acceleration ended, analyze orientation, and if the orientation revealed the user to be horizontal with the ground or, in the case of the user not being orientated upright, after a period of relative inactivity the user's orientation was still not upright, classify as a fall. The basic algorithm design for carrying this out proceeds as follows:

1) Look for a breech of a *Large Acceleration Threshold*
2) Wait until the large acceleration dissipates and we get relatively *Normal Acceleration*
3) Provide a short time interval *(around 12 seconds)* for the user to get acclimated
4) Analyze the user's orientation
5) If it is determined that the the user has *Orientation Horizontal with the Ground*, classify as a fall
6) If the user is not determined to have *Orientation Horizontal with the Ground*, but does have *Orientation Designated as Deviating from Uprightness*, provide another short time interval for the user to get acclimated
7) If, after a *Period of Inactivity Length*, the user still has *Orientation Designated as Deviating from Uprightness*, classify as a fall

The parameters to test in this case were *Large Accleration Threshold*, *Normal Acceleration*, and *Orientation Horizontal with the Ground*, *Period of Inactivity Length*, and *Orientation Designated as Deviating from Uprightness*.

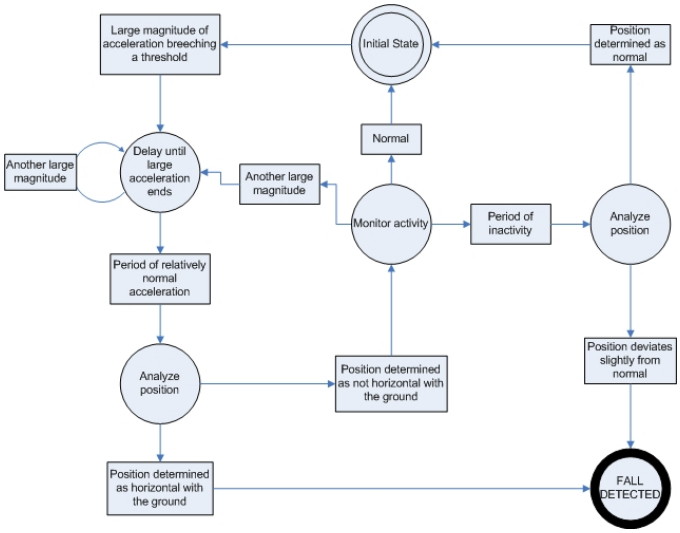


Fig. 9. Advanced Magnitude Algorithm

C. Testing

1) Testing Protocol: With the various changes in the algorithm, we realize that it is crucial to conduct the same type of testing in order to ensure that there is consistency in the detection schemes and that we are not losing fall-detection accuracy. By using common scenarios to test the changes in the algorithm, we can determine whether or not our changes actually improved or hurt our fall-detection success. According to a study by SignalQuest on Falls in the Elderly, 9% of falls were preceded by a loss of consciousness; 39% of the falls occurred while walking; 20% of the falls occurred while ascending or descending stairs or curbs; 24% occurred while transferring into or out of a bed or chair; 13% were caused by turning or reaching; 12% occurred while engaging in high risk behavior such as standing on a chair, running, or climbing [11]. By taking these statistics into account and by analyzing previous data and test results for similarities, we have developed a comprehensive list of various falls and nonfalls. Although there are many ways in which a user can fall, most falls can be classified into general categories. Three general categories devised are Simple Falls (falls in which the acceleration behavior shows relatively clearly that a fall has taken place and which the user ends up lying horizontal to the ground); Complex Falls (falls in which the acceleration behavior is somewhat complex and/or the user may end up in a non-horizontal position with the floor), and Non-Falls (actions in which the acceleration behavior is such that a false positive may occur). By providing exhaustive cases from each category to thoroughly test, we can then rest assured that most falls in that category will result in a similar outcome. From this list of falls, Garrett Brown and Rustom Dessai individually produced a data set for each fall.

D. Real World Orientation Analysis

As another idea which goes slightly behind the simplicity needed for a fall-detection scheme is a three-dimensional real world orientation model of the user which would be continuously updated in real time. By using spherical values computed from the acceleration data, it is possible to observe changes in the user's orientation with respect to a real world perspective. To correctly use the acceleration values to constantly transform the position matrix, it was necessary to first eliminate noise from the fall-sensor data. To do this we conducted various tests to create a low-pass filter with the specifcations to reduce most noise but still retain all the interesting acceleration properties. Based on the loss of noise versus loss of data ration, we decided on a second order lowpass digital Butterworth filter with a cuttof frequency of 0.04 (see Fig. 10). From here we have began an analysis of the methods of using the filtered data in trying to correctly orient the real world position.

V. RESULTS

A. Algorithm Refinement

From our initial idea of observing changes in angles in conjunction with a crossing of a set acceleration threshold within the same time interval, we developed the Conjoined Angle Change and Magnitude Detection Algorithm. Upon testing this algorithm with various parameters, the results were unsatisfactory (see Fig. 11). To avoid the many intricacies involved with angle changes, we decided to attempt a simpler idea with magnitude detection.

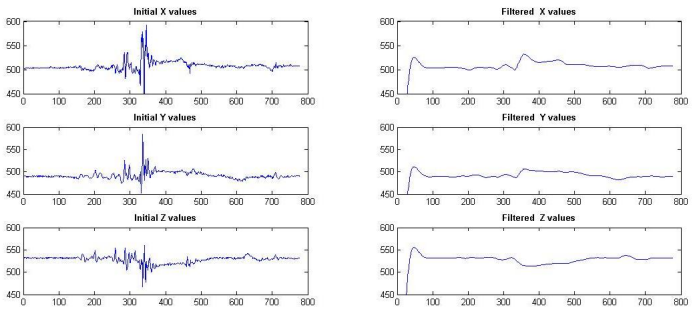


Fig. 10. Effects of applying filter to the data (*Note: although the noise and sharp peaks are removed, the behaviour of acceleration data stays the same)*

Conjoined Angle and Magnitude Detection Time Interval = 20 cycles(1/2 sec)	Simple Falls 51	Complex Falls 8	Total Falls 59	Non-Falls 18
Large Acceleration Threshold: **2G** Significant Angle Change:**π/3**	31	6	37	11
Large Acceleration Threshold: **1.6G** Significant Angle Change: **π/3**	39	8	47	12
Large Acceleration Threshold: **2.6G** Significant Angle Change: **π/3**	29	4	33	9
Large Acceleration Threshold: **2G** Significant Angle Change: **π/6**	31	6	37	11
Large Acceleration Threshold: **1.6G** Significant Angle Change: **π/6**	39	8	47	12
Large Acceleration Threshold: **2.6G** Significant Angle Change: **π/6**	29	4	33	9

Fig. 11. Conjoined Angle Change and Magnitude Detection Algorithm Statistics

1) Magnitude Detection: Although the simplicity of acceleration magnitude detection algorithms seems to undermine its potential, through testing and alterations of parameters we have found that the idea works suprisingly well. The Simple Magnitude Detection idea was much more effective than the Conjoined Angle Change and Magnitude Detection Algorithm (see Fig. 12). By varying parameters we were consistently detecting Simple Falls while avoiding False Positives. The only weakness we found was in Complex Falls or falls in which the user did not end up oriented horizontally with the ground.

To account for these cases, we altered the Simple Fall Detection idea. By utilizing more specific orientation analysis coupled with a period of inactiviy, we were much more effective at detecting Complex Falls while still avoiding most False Positives (see Fig. 13).

Due to the fact that some falls and non-fall behaviour are still too challenging to be handled correctly by this method, however, more work may need to be done on a more advanced fall-detection technique. For many cases, the magnitude detection algorithms will carry out very satisfactorily, but a more enhanced catch-all algorithm may also be used in conjunction in order to further increase the accuracy of fall-detection.

Simple Magnitude Detection Normal Acceleration = **Initial Calibration Values** Orientation Horizontal with the Ground = **(InitialZ – %G)**	Simple Falls 103	Complex Falls 20	Total Falls 123	Non-Falls 36
Large Acceleration Threshold: **1/2G** Ground Orientation Value: **.8G**	91	1	92	2
Large Acceleration Threshold: **1/2G** Ground Orientation Value: **.7G**	92	3	95	2
Large Acceleration Threshold: **1/2G** Ground Orientation Value: **.6G**	94	7	101	2
Large Acceleration Threshold: **1G** Ground Orientation Value: **.7G**	91	3	94	2
Large Acceleration Threshold: **1G** Ground Orientation Value: **.6G**	94	4	98	2
Large Acceleration Threshold: **1.1G** Ground Orientation Value: **.6G**	94	4	98	2
Large Acceleration Threshold: **1.2G** Ground Orientation Value: **.6G**	88	4	92	2
Large Acceleration Threshold: **1.5G** Ground Orientation Value: **.6G**	76	3	79	2

Fig. 12. Simple Magnitude Detection Algorithm Statistics

Advanced Magnitude Detection Normal Acceleration = **Initial Calibration Values** Orientation Horizontal with the Ground = **(InitialZ – .6G)** Period of Inactivity Length = **160 cycles** Orientation Designated as Deviating from Uprightness = **(InitialZ – .%G)**	Simple Falls 103	Complex Falls 20	Total Falls 123	Non-Falls 36
Large Acceleration Threshold: **1.1G** Normal Acceleration Threshold: **1.2G** Upright Deviation Orientation Value: **.1G**	96	17	113	2
Large Acceleration Threshold: **1.2G** Normal Acceleration Threshold: **1.2G** Upright Deviation Orientation Value: **.1G**	96	20	116	2
Large Acceleration Threshold: **1.24G** Normal Acceleration Threshold: **1.1G** Upright Deviation Orientation Value: **.1G**	96	16	112	2
Large Acceleration Threshold: **1.24G** Normal Acceleration Threshold: **1.2G** Upright Deviation Orientation Value: **..1G**	96	20	116	2
Large Acceleration Threshold: **1.24G** Normal Acceleration Threshold: **1.2G** Upright Deviation Orientation Value: **..2G**	94	20	114	2
Large Acceleration Threshold: **1.24G** Normal Acceleration Threshold: **1.3G** Upright Deviation Orientation Value: **..1G**	96	20	116	2
Large Acceleration Threshold: **1.31G** Normal Acceleration Threshold: **1.2G** Upright Deviation Orientation Value: **.1G**	96	19	115	2
Large Acceleration Threshold: **1.41G** Normal Acceleration Threshold: **1.2G** Upright Deviation Orientation Value: **.1G**	95	19	114	2

Fig. 13. Advanced Magnitude Detection Algorithm Statistics

B. Pattern Recognition

1) Fall Patterns: In many types of falls, the acceleration seems to follow some type of pattern (see Fig. 14). This pattern can be difficult to recognize, however, due to different characteristics such as physical stature of the user, fall speed, distance fallen, obstacles encountered during fall (see Fig. 15), and ending orientation but there are still some accleration schematics which may be consistent enough to establish a satisfactory pattern.

2) Non-Fall Patterns: Beyond the realm of simply recognizing fall acceleration patterns, there also seems to be potential for establishing patterns to recongize other common activities such as walking (see Fig. 16 and 17), sitting, climbing stairs, etc.

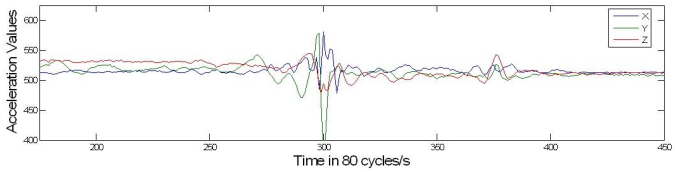


Fig. 14. Garrett falling backwards

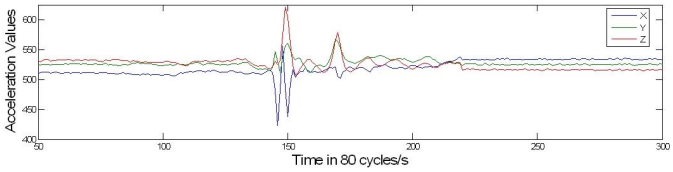


Fig. 15. Garrett falling rightwards after bouncing off a small object

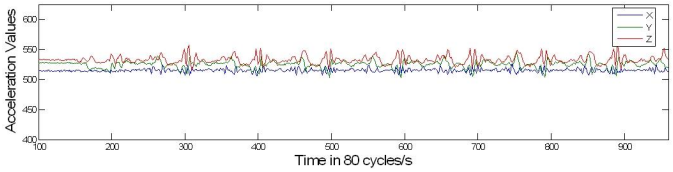


Fig. 16. Garrett walking normally

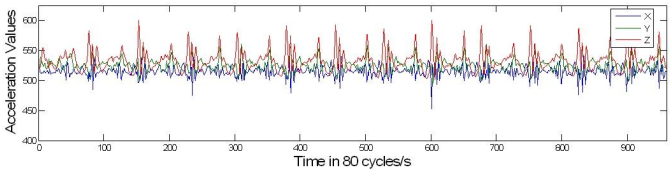


Fig. 17. Garrett walking briskly

VI. CONCLUSION

The idea of producing a simple, robust algorithm for fall-detection has produced very pleasing results. The algorithms developed thus far throughout this testing and analysis process are easily implementable onto the fall-sensor board. It is now known that many falls can be distinguished from non-falls by simply using acceleration magnitude and orientation analysis. Furthermore, there seems to be no noticeable accuracy difference between setting an acceleration threshold for each axis individually *(as in the case for the Simple Magnitude Detection Algorithm)* versus setting one magnitude threshold for the aceleration magnitude vector only *(as in the case for the Simple Magnitude Detection Algorithm)*. Functionality for communication between the fall-sensor board and the larger sensor network has also been developed satisfactorily. At the current stage of this project, based on controlled testing, the fall-sensor board can autonomously detect around 90% of all falls with only a 5% False Positive rate and then independently connect to a laptop or PC.

A. Future Ideas

1) Real World Perspective Analysis: Continuing with the idea of Real World Orientation Analysis posed in the result section, there is a lot of potential in terms of a model of the user with a real world perspective. Going beyond just simply knowing how the user is orientated in the real world is the idea that possibly certain changes that occur in the real world perspective can be determined to result from a fall. With stringent transformation procedures and close analysis, it is possible to eventually define transformation characteristics specific enough to distinguish falls from nonfalls. This idea could potentially lead to more defined observation and detection algorithms and hopefully more accuracy in terms of fall detection.

2) Least Mean Squares Adaptive Learning Algorithm: Going beyond the analysis of magnitude-detection algorithms discussed in this paper, there is a promising idea of using an adaptive learning algorithm to begin constructing a pattern recognizing fall-detection algorithm. As mentioned in the results section, there does appear to be a pattern in a majority of fall types. By using a strong analytical method, such as the least mean squares method, we can sort of train a filter to produce a given output from a given input. If we have an input signal and a known intended output, such as a fall-detected signal, the algorithm uses a least-squares approach to tune its own parameters such that it minimizes the error between its processed output and the desired output. Across several iterations, this should construct the filter such that it will produce an intended output for any given input. In theory, any type of fall then should have the basic pattern as our filter was trained to recognize and should then produce a strong correlation in pattern resemblence resuting in a fall-detection.

3) Acceleration Correlative Movement Library: Another hopeful idea is that there will be enough room on the fall sensor to begin a library of various actions. Eventually, with enough analysis of various actions, the fall sensor may eventually be able to go beyond simply detecting falls, but also provide continuous monitoring on the user's actions and condition. A possibility for doing this takes the same basic approach as the previous idea with the least means squares adaptive algorithm. If enough data is processed into separate initial filters, it may be possible to produce a filter for recognizing not only a fall, but also other actions, such as walking, running, sitting down, etc. Then, as data is read in from a sensor on the user, it can be passed through all the specific filters and its action then be designated by the filter which provides the strongest pattern correlation, thereby providing a means to monitor and predict a user's actions with a strong degree of certainty.

4) Extended Integrated Wireless Functionality: The next generation of the fall-sensor board will undoubtedly have more funcionality than the current board. There is the hope that in the near future, a Bluetooth device will be able to fit directly onto the board in order to provide the functionality necessary to use a Bluetooth interface in a much more compact and manageable form. The Bluetooth Server which now searches for the connection seeking signal provided by the fall sensor can be further developed to take user specified actions such as turning on a streaming camera, reading in data from other sensors, sending an e-mail or fax, or possibly even sounding some type of alarm. The increasing popularity of integrated wireless networks and functionality ensure that there will soon

be much more potential with respect to the capabilities of the sensors themselves and of the SensorNet network as a whole.

VII. Acknowledgments

The author would like to express his gratitude to his UC Berkeley faculty mentor Dr. Ruzena Bajcsy, project mentor Dr. Mikael Eklund, graduate student mentor Thomas Hansen, previous researchers Jay Chen and Karric Kwong, fellow researchers Rustom Dessai, Albert Chang and Andrew Chekerylla, the SUPERB-IT Program, the University of California at Berkeley, and the National Science Foundation.

References

[1] R. Bajcsy, J. Chen, K. Kwong, D. Chang, and J. Luk, *Fall detection using wireless sensor networks*, in *submitted to the 27th Annual International Conference of the EMBS*, September 2005

[2] M. Eklund, T. Hansen, S. Sastry, *Information Technology for Assisted Living at Home: building a wireless infrastructure for assisted living*, in *submitted to the 27th Annual International Conference of the EMBS*, September 2005

[3] M. Eklund, *Executive Summary: Information Technology for Assisted Living at Home (ITALH)*

[4] Loyola University Health System, *Don't Let a Fall Be Your Last Trip*, Available from URL: http://www.luhs.org/depts/injprev/Falls/adult.html [Cited 25 July 2005]

[5] E. Weisstein, *Cartesian Coordinates*, Available from Mathworld-A Wolfram Web Resource URL: http://mathworld.wolfram.com/CartesianCoordinates.html

[6] E. Weisstein, *Spherical Coordinates*, Available from Mathworld-A Wolfram Web Resource URL: http://mathworld.wolfram.com/SphericalCoordinates.html

[7] E. Weisstein, *Dot Product*, Available from Mathworld-A Wolfram Web Resource URL: http://mathworld.wolfram.com/DotProduct.html

[8] British Columbia Ministry of Health Planning Office of the Provincial Health Officer, *Prevention of Falls and Injuries Among the Elderly: A Special Report From the Office of the Provincial Health Officer*, January 2004, Available from URL: http://www.healthservices.gov.bc.ca/pho/special.html [Cited 26 July 2005]

[9] J. Chen, K. Kwong, *IvySpring04*

[10] M. Chen, F. Cheng, R. Gudavalli, *Precision and Accuracy in an Indoor Localization System*, December 2003

[11] J. Shea, *An Investigation of Falls in the Elderly*, Available from URL: http://www.signalquest.com/master%20frameset.html?undefined [Cited 25 July 2005]

This research was conducted by Garrett Brown at UC Berkeley in the Summer of 2005 through the SUPERB-IT research program. The author is currently a senior at the University of Michigan majoring in Computer Science. He aspires to come back UC Berkeley in fall 2006 for graduate school and continue pursuing his research interests in Human-Computer Interaction and Artificial Intelligence.