

A Fall Detection Module for Assisted Living

A dissertation

*submitted in partial fulfilment of
the requirements for the degree of*

Master of Technology

in

Electrical Engineering

by

Yogesh G. Gholap

(143073001)

under the supervision of

Prof. P. C. Pandey



Electronics System Group
Department of Electrical Engineering
Indian Institute of Technology Bombay
June 2016

Page intentionally left blank.

Indian Institute of Technology Bombay

M. Tech. Dissertation Approval

This dissertation entitled "**A Fall Detection Module for Assisted Living**" by **Yogesh G. Gholap** (Roll No. **143073001**) is approved, after the successful completion of *viva voce* examination, for the award of the degree of **Master of Technology in Electrical Engineering**.

Supervisor

.....
(Prof. P. C. Pandey)

Examiners

.....
(Prof. Joseph John)

.....
(Prof. L. R. Subramanyan)

Chairperson

.....
(Prof. D. K. Sharma)

Date: July 4, 2016

Place: Mumbai

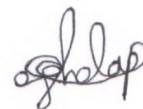
Page intentionally left blank.

Declaration

I declare that this dissertation represents my words and where ideas and words are taken from others, I have adequately cited and referenced the original sources. I declare that I have adhered to all principles of academic honesty and integrity and I have not misrepresented or fabricated or falsified any idea/data/facts/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Date: July 4, 2016

Place: Mumbai



Yogesh G. Gholap

Roll No.: 143073001

Abstract

Fall detection devices assist elderly persons in their activities of daily life. A fall detection module, using an integrated sensor with tri-axial accelerometer, tri-axial gyroscope, and tri-axial magnetometer and wireless connectivity, is developed. It is a microcontroller-based module with Bluetooth interface for wireless alert, operation control, and data transfer. After studying the signals for different orientations of the device during simulated falls, a signal processing technique using accelerometer outputs is developed and tested for orientation-independent discrimination of fall from activities of daily life. The technique is based on multi-dimensional decomposition of the output of the tri-axial accelerometer and finding the envelope of the deviation from the baseline for each of the components, and amplitude-deviation thresholding of the maximum of the deviation envelopes. The real-time implementation of the technique is carried out on the microcontroller for providing wireless alert on fall detection. It continuously monitors and records all sensor outputs at sampling frequency of 100 Hz in an internal flash memory for up to over an hour. The recorded data can be wirelessly transferred for analysis of physical activity of the person wearing the device. With lower sampling frequency, the device can be used for larger duration recording as needed for actigraphy.

CONTENTS

Abstract	i
List of abbreviations	iv
List of figures	v
List of tables	vii
 Chapters	
1. Introduction	1
1.1 Background	1
1.2 Project objective	1
1.3 Dissertation outline	2
2. Literature survey	3
2.1 Introduction	3
2.2 Signal processing for fall detection	3
2.2.1 Image processing based fall detection	3
2.2.2 Acoustic and vibration sensor based fall detection	3
2.2.3 Wearable sensor based method	3
2.3 Fall detection algorithm for wearable sensor	5
2.4 Design considerations for the hardware	6
2.5 Inertial sensor based fall detection developed earlier at IIT Bombay	7
3. Hardware design of fall detection module	10
3.1 Introduction	10
3.2 Sensor	10
3.3 Bluetooth transceiver	12
3.4 Flash memory	13
3.5 Microcontroller	14
3.6 Power supply and battery charger	16
3.7 PCB design	17
4. Hardware testing and software for data acquisition	18
4.1 Introduction	18
4.2 Programs for testing the hardware blocks	18
4.3 Programs for data acquisition	19
4.3.1 System initialization	20
4.3.2 Sample-by-sample mode data acquisition	21
4.3.3 Interrupt based burst-mode data acquisition	22

4.3.4	Polling based burst-mode data acquisition	23
4.4	GUI for data acquisition	23
4.5	Magnetometer calibration and testing	25
5.	Signal processing for real-time fall detection	29
5.1	Introduction	29
5.2	Test setups for signal acquisition	29
5.3	Investigations for signal processing	30
5.3.1	Deviation detector for discrimination of ADL and fall	32
5.3.2	Testing effectiveness of multidimensional decomposition for fall detection	35
5.3.3	Median filter using dynamic quantile tracking	36
5.3.4	Median filter using quick sort for true median	37
5.3.5	Squaring of input signal for contrast enhancement	38
5.4	Implementation for real-time processing	39
5.5	Testing of the device with real-time processing for fall detection	40
6.	Summary and further work	42
 Appendices		
A.	Component list	44
B.	Schematic diagram of FDM	45
C.	PCB layout	47
D.	Images of FDM prototype	49
E.	GIU developed for DAQ	50
F.	Plots of ADL data	51
G.	Plots of fall data	55
H.	Commercially available systems for fall detection	60
 References		61
 Acknowledgements		64
 Author's resume		65

LIST OF ABBREVIATIONS

Abbreviation	Explanation
ADC	analog-to-digital converter
ADL	activities of daily life
DAQ	data acquisition
DMP	digital motion processor
DPU	data processing unit
FDM	fall detection module
FIFO	first in first out
GUI	graphical user interface
I2C	inter integrated circuit
IMU	inertial measurement unit
I/O	input/output
IRQ	interrupt request
ISM	inertial sensing module
ISR	interrupt service routine
MCM	multi-chip module
MEMS	micro electro mechanical systems
MISO	master in slave out
MOSI	master out slave in
OSCO	oscillator out
PC	personal computer
PCB	printed circuit board
PIC	peripheral interface controller
RAM	random access memory
SCK	serial clock
SDA	serial data
SEMG	surface electromyograms
SPI	serial peripheral interface
UART	universal asynchronous receiver transmitter
USB	universal serial bus

LIST OF FIGURES

Figure	Caption	Page
2.1	Trunk resultant vector signal as given in [2]	5
2.2	Block diagram of the Inertial Sensing Module developed by Kumar [28]	8
3.1	Block diagram of the FDM	10
3.2	(a) Axis orientation of the accelerometer and gyroscope sensors in the integrated sensor MPU-9250 (b) Axis orientation of the magnetometer	11
3.3	Interfacing of integrated sensor to the microcontroller	11
3.4	Serial interface of the Bluetooth wireless transceiver and microcontroller	13
3.5	Memory interfacing with microcontroller	14
3.6	The microcontroller pin connections	16
3.7	Battery charger and power supply using 3.0 V LDO	17
4.1	Flowchart for initial configuration setting of programs	19
4.2	Flowchart for UART interrupt logic to control FDM operations	20
4.3	Sensor ISR in “Sample-by-sample” program	21
4.4	Sensor ISR in “Burst-mode interrupt” program	22
4.5	Data acquisition in “Burst-mode polling” program	24
4.6	GUI to control FDM	25
4.7	Magnetometer reading along X and Y axis for 360° rotation of FDM	26
4.8	Magnetometer data before and after calibration	27
4.9	Magnetometer readings for different magnets	28
5.1	FDM fixed at top of stick to record simulated falls	29
5.2	Wooden box fabricated for simulated fall with different orientations of FDM	30
5.3	Sensor outputs and corresponding decompositions for the running activity	31
5.4	Sensor outputs and corresponding decompositions for simulated fall with device orientation of yaw 0° and pitch 0°	32
5.5	Signal processing for detection of deviation envelope of signal component	33
5.6	Effect of different lengths of median filter on baseline and deviation from baseline, for the component a_x in Figure 5.4.	33
5.7	Example of signal processing results: input signal, estimated baseline, deviation from baseline, absolute deviation, deviation envelope, for the component a_x in Figure 5.4.	34

5.8	Deviation envelopes of the 11 decompositions of the accelerometer output and the corresponding maximum, for the fall data as in Figure 5.4.	34
5.9	Comparison of true median and median with dynamic quantile tracking, for the fall data in Figure 5.4.	36
5.10	Flowchart for real-time median filter using quick sort.	38
5.11	Example of signal processing results with squaring of the input signals for running and fall as in Figures 5.3 and 5.4, respectively.	39
5.12	Revised signal processing for detection of deviation envelope	39
5.13	Signal processing for detection of maximum deviation envelope.	40
5.14	Flowchart of threshold-based real-time fall detection algorithm	41
B.1	Main schematic	45
B.2	Power supply and battery charger	46
C.1	Top layer of PCB	47
C.2	Bottom layer of PCB	48
D.1	Images of FDM prototype	49
E.1	GUI Screenshot developed for DAQ in MATLAB	50
F.1	Plots of ADL data: running	51
F.2	Plots of ADL data: jumping	52
F.3	Plots of ADL data: sitting down on a chair and getting up	52
F.4	Plots of ADL data: sitting down on a chair and getting up with jerk	53
F.5	Plots of ADL data: hopping	53
F.6	Plots of ADL data: walking up stairs and down stairs	54
F.7	Plots of ADL data: walking	54
G.1	Plots of data for simulated fall using the waist-height stick, with the device facing to the north and falling on the right with twist.	55
G.2	Plots of data for simulated fall using the waist-height stick, with the device facing to the north and falling backward.	56
G.3	Plots of data for simulated fall using the waist-height stick, with the device facing to the north and falling forward.	56
G.4	Plots of data for simulated fall using the waist-height stick, with the device facing to the north and falling on the left with twist.	57
G.5	Plots of data for simulated fall using the waist-height stick, with the device facing to the east and falling on the right with twist.	57
G.6	Plots of data for simulated fall using the waist-height stick, with the device facing to the east and falling backward with twist.	58
G.7	Plots of data for simulated fall using the waist-height stick, with the device facing to the east and falling forward with twist.	58
G.8	Plots of data for simulated fall using the waist-height stick, with the device facing to the east and falling on the left with twist.	59

LIST OF TABLES

Table	Caption	Page
2.1	Summary of fall detection methods [1]	4
2.2	A comparison of IMUs	6
2.3	A comparison of microcontrollers	7
2.4	Comparison of RF communication modules	7
2.5	Components used in the inertial sensor module developed by Kumar [28]	8
3.1	Assignment of port pins of the microcontroller U4	15
3.2	Supply voltage and current of various devices	16
5.1	Variables which best capture fall out of x , y , z , c_{xy} , c_{yz} , c_{zx} , c_{xyz} , m_{xyz} , m_{xy} , m_{yz} , m_{zx}	35
A.1	Component list of the FDM	44
H.1	Comparison of commercial products [15]	60

Chapter 1

INTRODUCTION

1.1 Background

Advancements in healthcare have led to increase in average life span and hence have put a challenge to efficiently cater to the needs of the elderly persons. Many elderly persons may not have good control over activities of daily life (ADL) like walking, sitting down, standing up, lying down etc, due to reduced muscular strength (especially in legs), diminished vision, and lack of reflexes. It is not feasible for caretakers to provide them with continuous personal assistance. For such persons, recovery from any injuries to muscles or bones is very gradual. Early detection of such injuries is important as delay in treatment can prove fatal. Solutions need to be developed to assist the elderly persons in independent living and fall detection is one of the major supports that can provide them confidence for ADL.

The commonly used fall detection techniques are (i) computer vision-based methods, (ii) acoustic and ambient sensor-based methods, and (iii) wearable sensor-based methods [1]. Compared to other methods, the wearable sensor approach is more convenient, cheaper, and does not introduce any privacy related issues. It is based on wearable modules, involving interfacing of sensors like accelerometer, gyroscope, magnetometer, tilt sensor and surface electromyograms (SEMG) with processor and an alert mechanism [1].

For effectiveness of the wearable module, hardware design is important to get high accuracy, good battery life, and compact size. The convenience of the user and aesthetics should also be considered. The device can be placed on the wrist as a watch, or can be worn around the neck, or worn on the clothing using a belt or a clip [2]. Several algorithms have been developed to process the real-time sensor data. Most of the fall detectors use threshold based logic, where certain limit is set on the parameters obtained from the sensor. If these parameters cross the permissible threshold then fall condition is detected. The main challenge lies in detecting every fall and not generating alarm during ADL, i.e. 100% sensitivity and specificity.

1.2 Project objective

A fall detection device reported by Kumar and Pandey [7] used an integrated sensor with tri-axial accelerometer and tri-axial gyroscope and a fall detection algorithm which monitored variations in acceleration along 7 different directions. The aim of the current project is to develop a body-worn wireless fall detection module, using a combination of tri-axial accelerometer, gyroscope, and magnetometer, and to investigate processing of the sensor

data for fall detection with improved sensitivity and specificity. The designed module is battery powered with USB charging.

1.3 Dissertation outline

The second chapter provides literature survey of hardware and signal processing for fall detection. The third chapter describes the hardware design of the fall detection device using MPU-9250 sensor having tri-axial accelerometer, gyroscope, and magnetometer. The fourth chapter describes the testing of the hardware and development of associated software. The fifth chapter presents the signal processing techniques to discriminate between ADL and fall, its implementation for real-time processing, and test results. The last chapter provides the summary and conclusions.

Chapter 2

LITERATURE SURVEY

2.1 Introduction

In this chapter, the first section provides an overview of sensing processes for fall detection. The second section describes the threshold based fall detection. The design considerations for the fall detection are presented in the third section and a device developed by Kumar [28] is described in the last section.

2.2 Sensing processes for fall detection

There are various sensors and sensing processes for fall detection, the predominant ones being based on image processing, acoustic and vibration monitoring, and inertial sensing, as summarized in Table 2.1, and described in the following subsections.

2.2.1 *Image processing based fall detection*

This method is used for indoor monitoring, using multiple cameras placed in the room. With the help of image processing application, the posture of the person is monitored by an elliptical approximation of body [3]. The current posture is extracted and its parameters are compared with reference database. Another method is to detect the centroid of the body and to find the distance of centroid from floor plane [1]. These methods have been reported to provide sensitivity and specificity as good as 94.4% and 98.8% respectively.

A thermal imaging system using infrared-array has been reported [9] for detecting motion and also inactivity for a preset time. Infrared sensor array is cheaper than camera based detection. The system uses fall detection algorithm based on vertical fall velocity for reducing the false alarms.

2.2.2 *Acoustic and vibration sensor based fall detection*

It is possible to detect and differentiate the human fall activity by monitoring the sound and vibrations produced after a fall. Such devices are more economical as compared to those based on camera and image processing. However they require very complex signal processing algorithm for high specificity [1].

2.2.3 *Wearable device based method*

This method uses a body worn device and is more economical and fairly accurate as compared to the earlier two methods. With advancement in micro-electromechanical sensors (MEMS) technology, inertial sensors with small size and low cost have become available.

Table 2.1: Summary of fall detection methods [1].

Traits	Fall Detection		
	Image processing	Acoustic and vibration	Wearable sensor
Space	Indoor	Indoor	Anywhere
Privacy	Violated	Protected	Protected
Cost	Expensive	Expensive	Moderate
Reported accuracy	97.3%	100%	99.92%
Deployment	Simple	Complex	Very simple

These devices monitor kinematic parameters like gravitational acceleration, angular position and angular velocity. Compact inertial measurement units (IMU) are available with tri-axial gyroscope and tri-axial accelerometer with signal conditioning units. Various other sensors that are used with inertial sensors as extra conformation for fall are tilt sensor [4], surface electromyography (SEMG) sensor [5], pressure bio-sensor [10], and magnetometer [11] [12].

Hwang et al. [4] used tilt sensor in addition to accelerometer for posture detection. Tilt sensor are digital sensors which detect whether the object is tilted beyond a certain threshold value. Above certain tilt threshold, values from accelerometer and gyroscope are recorded, and if these values cross the corresponding threshold, then fall is predicted. The accuracy of developed device is reported as 96.7%.

Cheng et al. [5] studied use of SEMG based biosensor to monitor muscular stress. These are non-invasive sensors placed on skin surface and provide information related to muscle activity and stress. Four such SEMG electrodes are placed along the lower limb muscles. Two tri-axial accelerometers are placed, one on the chest and other on the right thigh. From the output of these sensors and with the help of hidden Markov model (HMM) of human body motion, dynamic gait activities are recognized. They reported fall recognition accuracy as 98%.

Bianchi et al. [13] used barometric pressure sensors to sense altitude information to differentiate between various body positions like standing, sitting, sleeping etc. The pressure sensor having resolution of 1.5 Pa which corresponds to about 10 cm at sea level has been used. As fall occurs and the body loses the height, pressure starts increasing and hence it is an indication of fall.

Gallagher et al. [11] studied combination of accelerometer and magnetometer to decide the direction of fall e.g. whether person falls forward or on the back. This combination provides orientation information about body with respect to two directions, corresponding to earth's gravitational and magnetic fields. Because magnetometer is very sensitive to stray magnetic fields, it often tends to provide false data.

Li et al. [14] designed system with Telos-W mote placed at the waist of the user with accelerometer based detection and Zigbee wireless connectivity. The development board used

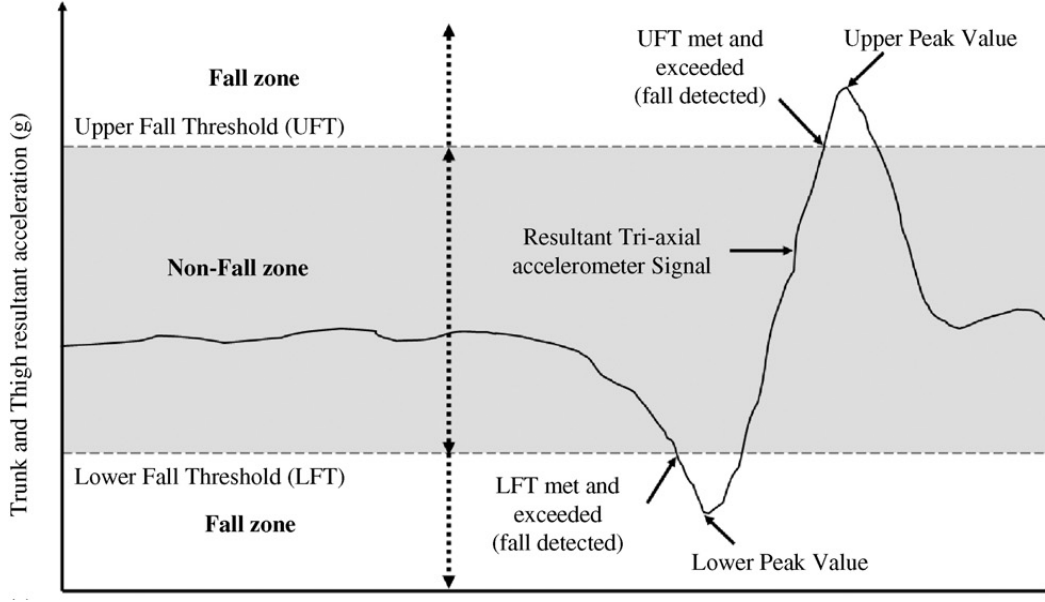


Figure 2.1: Trunk resultant vector signal as given in [2].

ADXL345 accelerometer, MSP 430 microcontroller, and CC1101 wireless. Algorithm used for fall detection is threshold based and works on data sampled at 50 Hz. The fall detection accuracy is reported as 92%

2.3 Fall detection algorithm for wearable sensor module

The processor of the wearable sensor module has to analyze the sensor data in real-time and detect the fall. The algorithms used may be broadly classified as (i) computation intensive algorithm, (ii) body posture-based algorithm, and (iii) multisensory strategy [5]. To predict posture, Kalman filter based rotational matrix can be used [11], but it requires higher processing power than that available in most microcontrollers.

In threshold based detection, the outputs from the accelerometer is used to get acceleration vector. A plot of sensor response during fall is shown in Figure 2.1 [2]. It is assumed that there are two threshold values within which ADLs occur. The values outside the thresholds are treated as fall. The detection can be further improved by adding additional threshold in time domain [7]. It is observed that during fall, acceleration vector value remains above threshold for certain duration. If it remains high for some minimum duration and then it starts decreasing, fall is detected. Otherwise the values are treated as ADL values and no alarm is generated

Placement of the sensor module should be selected to improve sensitivity and specificity. The sensor can be placed on wrist, forehead, thigh or trunk [8]. It has been reported that sensor placed at waist and head can provide high sensitivity and specificity. It is not comfortable to wear sensor on forehead. Sensor placed on wrist is convenient but has

Table 2.2: A comparison of IMUs.

Specification	MPU-9250 [19]	LSM330DLC [24]
Manufacturer	InvenSense	ST Microelectronics
Accelerometer	3-axis ($\pm 2g$ to $\pm 16g$)	3-axis ($\pm 2g$ to $\pm 16g$)
Gyroscope	3-axis (± 250 - $2000^\circ/\text{sec}$)	3-axis (± 250 - $2000^\circ/\text{sec}$)
Magnetometer	3-axis ($\pm 4800\mu\text{T}$)	None
ADC	16 bit for Acc & Gyro	12 bit Acc & 16 bit Gyro
Storage (FIFO)	512 Bytes	2 \times 32 \times 16 Bytes (32 samples)
Supply voltage	2.4 V to 3.6 V	2.4 V to 3.6 V
Gyroscope operating current	3.2 mA	6.1 mA
Accelerometer operating current	450 μA	11 μA
Magnetometer operating current	280 μA	-----

been reported to have very low specificity. Bourke et al. [2] showed that sensor placed on the trunk have higher specificity than that on thigh. Multiple sensor units can be used to improve the accuracy by placing them on different body parts. Though this method proves to be more accurate, it may be cumbersome for an elderly person to wear them [10].

2.4 Design considerations for the hardware

The hardware design should have high accuracy of sensor data, low power consumption, adequate data storage, and small size. Many available inertial measurement unit (IMU) devices have accelerometer, gyroscope, magnetometer, and temperature sensor, ADC, data storage, and processor as single package. Features of two IMU are summarized in Table 2.2. The sampling rate is user settable in these IMUs. A higher sampling rate for the sensor output may improve the accuracy of the fall detection, but it increases the power consumption and data storage requirement. Sampling rate of 50 Hz and 100 Hz are most commonly used and are considered to be adequate for posture tracking and 50 Hz as acceptable for fall detection [6].

Memory for data storage should have adequate data transfer speed and storage capacity. A microcontroller is used as the central control and processor to read data from sensor and store it to memory with appropriate time stamp. For power efficient operation, the microcontroller is kept in sleep mode for most of the time and data transfer is interrupt driven. Some of the microcontrollers available with this feature are Microchip PIC24, TI MSP430, ARM LPS2148 etc. Also adequate number of ports is required so as to interface various peripherals like IMU, memory, and wireless module. All above microcontrollers support I2C, SPI, and UART communication. Features of some microcontrollers are summarized in Table 2.3. For data communication between the sensor module and fall detection module (FDM), wireless communication is needed, which can be provided using infrared, Bluetooth, or Zigbee. The features of two RF communication modules are summarized in Table 2.4.

Table 2.3: A comparison of microcontrollers.

Specification	MSP-430 [27]	LPC2148 [26]	PIC 24 [19]
OEM	TI	Phillips	Microchip
ALU	16 bit	16 bit	16 bit
Operating voltage	1.8 V	3.0 V	2.0 V
Active power	3 mW	0.6 mW	1.3 mW
Sleep power	6 μ W	2 μ W	0.3 μ W
Communication	I2C, SPI, UART	I2C, SPI, UART	I2C, SPI, UART

Table 2.4: Comparison of RF communication modules.

Specification	CC-2420 [27]	RN42 [20]
Protocol	Zigbee 802.15.4	Bluetooth 802.15.1
Data rate	250 Kbps	240 Kbps
Transmit current	17 mA	30 mA
Supply voltage	1.8 V	3.3 V
Power consumption	40 mW	130 mW

2.5 Inertial sensor based fall detection developed earlier at IIT Bombay

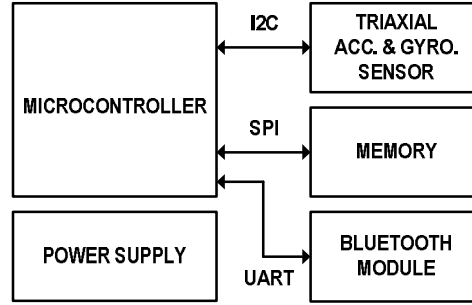
An accelerometer and gyroscope based fall detector was earlier developed by Kumar [28] at IIT Bombay. Its block diagram is shown in Figure 2.2 and its components are listed in Table 2.5. Integrated sensor “MPU-6000” (from InvenSense) is used which has on-chip tri-axis accelerometer and tri-axis gyroscope. The microcontroller is interfaced to the Bluetooth transceiver “RN42” (from Roving Network) for wireless connectivity for commands and data transfer. The flash memory “SST25VF064C” (from Microchip) of 8 MByte is used. To control and process the data microcontroller “PIC24FJ64GB004” (from Microchip) has been used. The module is operated at 3.3 V provided by 9 V battery and linear regulator “MCP1802” (from Microchip).

The sensor MPU-6000 has 16-bit ADC for each channel and I2C port for data transfer. The data can be output as 2-byte 16-bit signed integer or can be stored in on-chip 1024 byte buffer. Module is initialized for data acquisition rate of 100 Hz and ADC data from 3 axes for each accelerometer and gyroscope are taken every 10 ms and are stored in on-chip register/buffer. Thus 12 bytes of data are generated every 10 ms. The microcontroller configures various modules (accelerometer: $\pm 4g$, gyroscope: $\pm 500^\circ/s$ at sampling rate 100 Hz) and then goes into sleep mode. It can then only respond to the UART commands. A PC-based GUI (developed with MATLAB platform) is used to request for data or control the data acquisition modes. It can initiate data acquisition processes via sending appropriate command over Bluetooth transceiver.

On receipt of valid command, microprocessor enables IMU to start data acquisition in either of two modes viz. sample-by-sample data acquisition or burst-mode. In sample-by-

Table 2.5: Components used in the inertial sensor module developed by Kumar [28].

Device	OEM	Communication	Current	Data rate
IMU	Inven Sense MPU-6000	(I2C)	3.9 mA	400 kbps
Flash memory	Microchip SST25VF064C	(8 MB SPI)	12 mA	1 Mbps
Bluetooth module	Roving Network RN42	(UART)	30 mA	115 kbps
Microcontroller	Microchip PIC24FJ64GB004	I2C SPI UART	4.2 mA	400 kbps, 1 Mbps, 115 kbps

**Figure 2.2:** Block diagram of the Inertial Sensing Module developed by Kumar [28].

sample mode, timer interrupt is generated every 10 ms and 12-byte data are transferred to microcontroller and saved in its on-chip RAM buffer of 256 byte. Once this buffer is full then data is transferred to flash memory. This data can be then read by issuing another ‘data read’ command over Bluetooth transceiver. In interrupt driven burst-mode after every 10 ms, 12-byte data are stored in on-chip buffer of IMU. Thus total 85 samples are recorded in 850 ms. Once this on-chip buffer is full, sensor module generates the interrupt and controller comes out of sleep mode. Microcontroller then stores this data in flash memory. In this mode, the microcontroller gets about 850 ms to process the data for fall detection. Due to IMU buffer misalignment, one sample is always lost in this mode. In polling-based burst-mode, the microcontroller does not go in sleep mode after data acquisition is initiated. It keeps on checking sample count from IMU. Once desired number of samples, N (< 86 at sampling rate 100 Hz) are stored in IMU buffer, the data are read by the microcontroller over I2C bus and stored in the flash while IMU is still reading next set of samples. This method results in higher power consumption as the microcontroller is not in sleep mode. The count N is selected as 21, i.e., 252 bytes are stored at a time. Although the outputs from both accelerometer and gyroscope are recorded, only the accelerometer output is processed for fall detection.

Individual components $A_x(n), A_y(n), A_z(n)$ and the magnitude of the projection of the vector on xy, yz, and zx planes and vector magnitude used in fall detection are as the following:

$$A_{xy}(n) = \sqrt{A_x^2(n) + A_y^2(n)} \quad (2.1)$$

$$A_{yz}(n) = \sqrt{A_y^2(n) + A_z^2(n)} \quad (2.2)$$

$$A_{zx}(n) = \sqrt{A_z^2(n) + A_x^2(n)} \quad (2.3)$$

$$A_{xyz}(n) = \sqrt{A_x^2(n) + A_y^2(n) + A_z^2(n)} \quad (2.4)$$

It is assumed that a fall is likely to be indicated by a large change in a majority of these variables. Thus threshold detection on all these variables is likely to improve sensitivity and specificity as compared to that on $A_{xyz}(n)$ alone. A moving average for 100 samples is calculated for each of seven variables and difference between the current value and the moving average is calculated as the deviation $d_i(n)$. Now this difference is used to predict the fall depending on threshold values decided from experimental database θ_i . If $d_i(n)$ remain above threshold for duration greater than t_1 but less than t_2 then fall is predicted. Fall detection with threshold values of $\theta_i = 2g$, $t_1 = 250$ ms and $t_2 = 850$ ms provided satisfactory results [7]. Using polling based burst-mode to read sensor data, the real-time fall detection algorithm was implemented in the microcontroller.

Chapter 3

HARDWARE DESIGN OF FALL DETECTION MODULE

3.1 Introduction

The fall detection module (FDM) is designed as a wearable battery-powered device. Its block diagram is shown in Figure 3.1. It has integrated sensor, microcontroller, flash memory, Bluetooth transceiver, and chargeable battery. The integrated sensor has tri-axial accelerometer, tri-axial gyroscope, and tri-axial magnetometer along with the digital interface. The microcontroller reads the sampled data from the sensor and stores them in the flash memory. The microcontroller is also used for real-time processing of data for fall detection. The Bluetooth based wireless interface is provided so as to communicate with a computer or a mobile handset. The device can be used to store and transmit real-time values of all three sensors viz. accelerometer, gyroscope, and magnetometer and for alert generation after fall detection. The device operates on chargeable battery which can be charged using mini-USB connector. As the module is to be used as a wearable device, it should have low weight and small size. This chapter describes the hardware design.

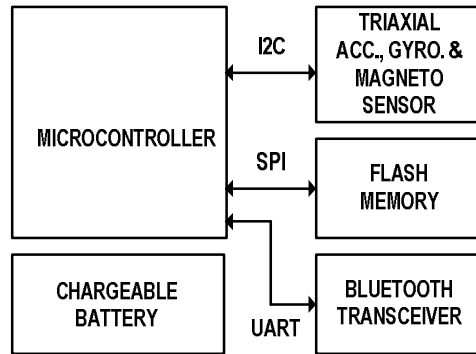


Figure 3.1: Block diagram of the FDM.

3.2 Sensor

The core of the module is the integrated sensor “MPU-9250” (from Microchip) [18], a 9-output motion tracking device consisting of 3 sub-sensors viz. tri-axial accelerometer, tri-axial gyroscope, and tri-axial magnetometer. Each sub-sensor can sense along three mutually perpendicular axes as shown in Figure 3.2. It is an integration of two blocks in single package: (i) accelerometer and gyroscope with signal acquisition and controller and (ii) magnetometer. The analog outputs of each sensor along the three axes are converted into 16-bit signed values using the on-chip ADC. The sampling rate can be set from 3.9 Hz to 8 kHz for accelerometer and gyroscope. The magnetometer output can be sampled either at 8 Hz or 100 Hz. The sampling rate of 100 Hz has been reported to be sufficient for fall detection using

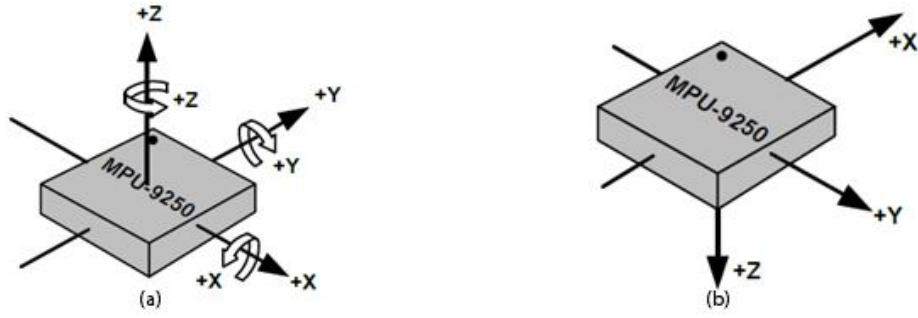


Figure 3.2: (a) Axis orientation of the accelerometer and gyroscope sensors in the integrated sensor MPU-9250 (b) Axis orientation of the magnetometer.

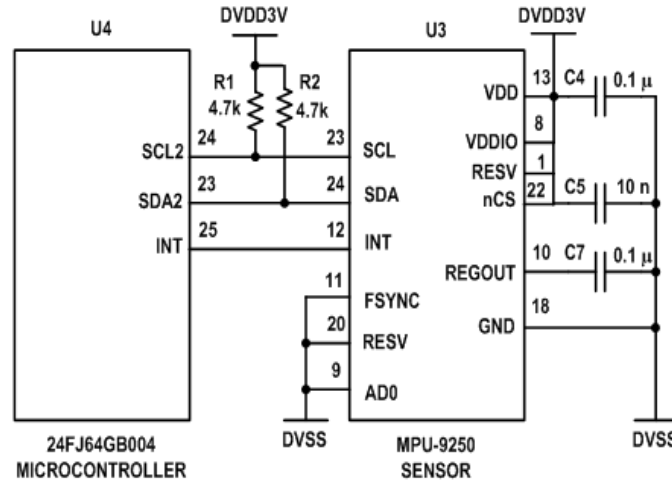


Figure 3.3: Interfacing of the integrated sensor to the microcontroller.

accelerometer [6]. Hence we use it for all the three sensors. Thus the sensor module provides 18 bytes of data (2 bytes for each axis of the three sensors) at 100 Hz. The ‘data ready’ interrupt is generated on buffer full and it can be used to read the data from the sensor. The full-scale range for accelerometer can be configured as $\pm 2g$, $\pm 4g$, $\pm 8g$, $\pm 16g$, and that for gyroscope as ± 250 , ± 500 , ± 1000 , ± 2000 $^{\circ}/s$. The full-scale range for magnetometer is ± 4800 μT . The sensor operates at 2.4 – 3.6 V and consumes about 4 mA current in normal operating mode. It is available in 3 mm \times 3 mm \times 1 mm QFN package.

The interfacing of the sensor chip U3 to the microcontroller U4 is shown in Figure 3.3. The sensor has serial communication with I2C (400 KHz) and SPI (1 MHz) protocol. I2C needs 2 lines (SCK, SDA) for communication, while SPI needs 4 lines (SCK, MOSI, MISO, CS). SPI does not have a built-in mechanism for acknowledgement and addressing. Although, I2C has a lower data rate than SPI, its data rate 400 kHz is sufficient for transferring the data for sampling rate of up to 1 kHz as needed in our application. Hence, the sensor is interfaced to the microcontroller using I2C. The SDA and SCL pins for I2C communication are open

drain and therefore pull-up resistors R1 and R2 of 4.7 k Ω are provided. Capacitor C7 (0.1 μ F) is connected to REGOUT as filter capacitor for the internal regulator of the chip.

The sensor “MPU-9250” consists of two dies, one being MPU-6515 having tri-axial accelerometer and tri-axial gyroscope while the other being AK8963 tri-axial magnetometer. MPU-6515 acts as master core and AK8963 is interfaced as I2C slave over auxiliary I2C port. This auxiliary I2C port can be connected to main I2C bus using bypass mode. In this mode, the clock frequency is restricted to 400 kHz. The I2C bus can be operated at 1 MHz clock frequency if bypass mode is off. If both cores are to be accessed, bypass mode should be on and maximum clock frequency supported is 400 kHz.

The sensor module has an on-chip digital motion processor (DMP) which can be used for reducing the computation requirement of the host processor. At each sampling instant, the 16-bit ADC values of all 9 outputs are stored to their corresponding output registers. These 18 bytes can be then transferred to FIFO. Sample-by-sample mode involves timer interrupt based 18-byte data transfer after each sampling interval. The sensor module has an on-chip 512-byte FIFO buffer. As the buffer gets filled, an interrupt is generated after which the data can be read in burst-mode. This mode permits a larger fraction of time on the microcontroller for processing of the acquired data. The FIFO buffer has a counter associated with it for tracking the number of bytes in the buffer. The counter value can be read at any time for carrying out polled data transfer from the buffer.

3.3 Bluetooth transceiver

The Bluetooth transceiver is used for wireless connectivity with various devices like laptop and mobile phones and it has 20 m range, generally sufficient for indoor application. For this purpose, the Bluetooth module “RN42” (from Roving Network) [20] is selected. It comes in 35-pin LCC package with dimension of 13.4 mm \times 25.8 mm \times 2.4 mm. The operating voltage is 3 – 3.6 V. It consumes 30 mA during data transfer and 3 mA otherwise. This chip can be interfaced over UART to communicate with the microcontroller over default baud rate of 115 kbps.

The data are transferred from the microcontroller U4 to the Bluetooth transceiver U2 serially using UART, as shown in Figure 3.4. The module has a programmable pin PIO7 to set the baud rate at power-on. If this pin is kept high at power-on the baud rate is 9600 bps, which cannot be changed later by programming. If it is low at power-on or is not connected the baud rate is 115 kbps and can be changed later to 1200, 2400, 4800, 9600, 19.2k, 28.8 k, 38.4 k, 57.6 k, 115 k, 230 k, 460 k, or 921 k by using configuration command SU in command mode. In our design, PIO7 is not connected and therefore the power-on baud rate is 115 kbps. It is not changed by programming, as this rate is considered to be sufficient for our application.

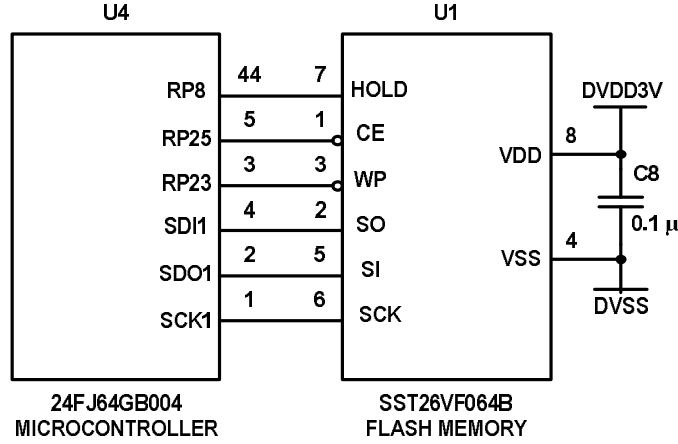


Figure 3.5: Memory interfacing with the microcontroller.

The SPI has been used for connecting the memory U1 to the microcontroller U4 as shown in Figure 3.5. Each sensor axis data are of 2 bytes and hence the 9-outputs (tri-axial accelerometer, tri-axial gyroscope and tri-axial magnetometer) are read as 18 bytes. Therefore, 419430 samples (8 M bytes / 18 bytes) can be stored in the memory. At a sampling frequency of 100 Hz, the memory can record data for 4194.3 s, i.e. 1 hour 10 minutes. In applications like actigraphy where sampling frequency of 10 Hz may be sufficient, the memory can store data for 11 hours 39 minutes. Capacitor C8 (0.1 μ F) serves as the supply decoupling capacitor. The pin WP is for hardware control of write protection while the pin HOLD is for pausing a serial sequence. HOLD pin also provides a power-on reset to the memory and can be used as a reset pin by clearing the EHL D (Enable Hold) bit of the memory.

3.5 Microcontroller

The microcontroller “PIC24fj64gb004” (from Microchip) [19], is a 16-bit microcontroller with 44 pin TQFP 10 mm \times 10 mm \times 1 mm package, It uses supply voltage of 2 – 3.6 V and consumes about 4.2 mA current while operating at 4 MIPS. As precision of the processor clock is not essential in our application, the system is designed to use the on-chip RC oscillator of 8 MHz which can be divided using 8-bit post-scalar to generate processor clock F_{OSC} . The controller takes two clock cycles to execute one instruction and thus its instruction cycle clock rate $F_{CY} = F_{OSC}/2$, i.e., 4 MHz. It provides serial communication options of I2C, SPI, and UART, and is USB OTG enabled. It provides reconfigurable peripheral pins which is a very helpful feature in optimizing the PCB layout.

Table 3.1: Assignment of port pins of the microcontroller U4.

Pin No.	Pin Label	Function
1	SCK1	Serial clock of SPI connected to serial clock of flash memory U1
2	SDO1	Serial data out of SPI connected to serial data in of flash memory U1
3	RP23	Digital output connected to write protect of U1 flash memory U1
4	SDI1	Serial data in of SPI connected to serial data out of flash memory U1
5	RP25	Digital output connected to chip enable of flash memory U1
23	SDA2	Serial data of I2C connected to serial data of sensor U3
24	SCL2	Serial clock of I2C connected to serial clock of sensor U3
25	INT	Digital input connected to interrupt of sensor U3
34	RA4	Digital output connected to reset of Bluetooth module U2
35	RA9	Digital output connected to PIO6 of Bluetooth module U2
36	RP19	UART Rx connected to Bluetooth U2
37	RP20	UART Tx connected to Bluetooth U2
38	RC5	Digital output connected to PIO4 of Bluetooth module U2
41	RB5	Digital output connected to PIO2 of Bluetooth module U2
43	RB7	Digital output connected to PIO3 of Bluetooth module U2
44	RP8	Digital output connected to HOLD of flash memory U1

The connections to the microcontroller U4 are shown in Figure 3.6. The power and inline programming connections are made as per the information in its datasheet. As the analog blocks of the microcontroller are not used, the AVDD and AVSS pins are shorted to DVDD and DVSS, respectively. Capacitors C1, C2, C3 (0.1 μ F, ceramic) serve as supply decoupling capacitors, each placed across DVDD-DVSS, within a distance not more than 6 mm from the pins. The DISVREG pin is connected to DVSS to enable the internal voltage regulator. To filter the output voltage of the on-chip voltage regulator, capacitor C6 (10 μ F) is connected between VCAP and DISVREG pins at a distance less than 6 mm from the pins. The connector CN1 is a 5-pin connector provided for inline programming and debugging. PGEC/PGED pins are used for in-circuit serial programming and debugging. The MCLR pin is used for device programming and debugging as well as device reset. A resistor R3 (100 Ω) is connected to MCLR pin to limit the current entering the pin, and to meet VIH and VIL specifications at that pin. Pins 23, 24, 25 (SDA2, SCL2, INT) are used for interfacing the sensor as described in Section 3.2 and shown in Figure 3.3. The flash memory U1 is interfaced to pins 1, 2, 3, 4, 5, 44 (SCK1, SDO1, RP23, SDI1, RP25, RP8) as described in

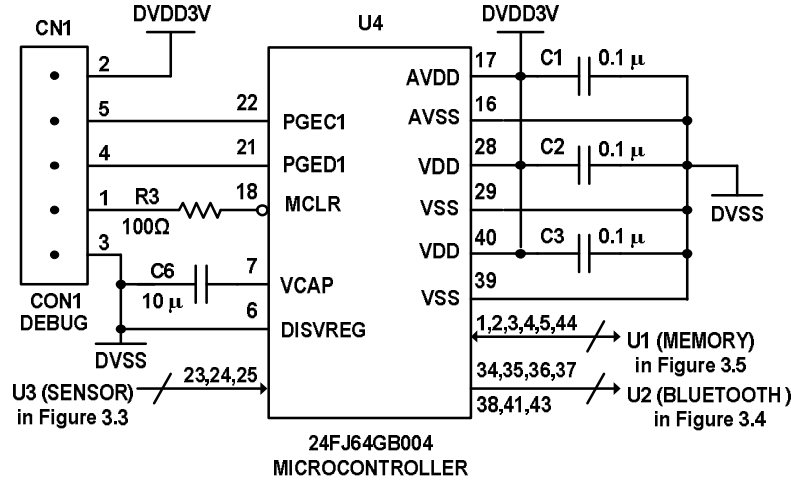


Figure 3.6: The microcontroller pin connections.

Table 3.2 Supply voltage and current of various devices.

Component	Minimum operating Voltage (V)	Maximum operating Voltage (V)	Maximum current drawn (mA)
Microcontroller U4 “PIC24”	2.0	3.6	4.2
Inertial sensor U3 “MPU 9250”	2.4	3.6	3.9
Bluetooth transceiver U2 “RN42”	3.0	3.6	30
Flash memory U1 “SST26”	2.6	3.6	25

Section 3.4 and shown in Figure 3.5. The Bluetooth transceiver is connected to the pins 34, 35, 36, 37, 38, 41, 43 (RA4, RA9, RP19, RP20, RC5, RB5, RB7) as described in Section 3.3 and shown in Figure 3.4. The assignment of the microcontroller port pins used for interfacing the sensor, the flash memory, and the Bluetooth module is given in Table 3.1.

3.6 Power supply and battery charger

The components have been selected such that they can be powered using a single regulator. The supply voltage and current requirements of the chips are given in Table 3.2. The Bluetooth transceiver requires at least 3 V and other chips can also work at this voltage. Thus whole system is designed to be powered with single 3 V supply. The low dropout (LDO) voltage regulator U6 “MCP-1802-3.0” (from Microchip) [22] is used. It has dropout voltage of 200 mV at 100 mA output current and can provide up to 300 mA.

The wearable module is powered using 3.7 V Li-ion rechargeable battery of capacity 1000 mAh. The maximum current consumption of the module is about 60 mA. Therefore the battery can last over 13 hours. The FDM is designed with onboard battery charger U5 “MCP73833” (from Microchip) [23]. A mini-USB port is provided for charging the battery

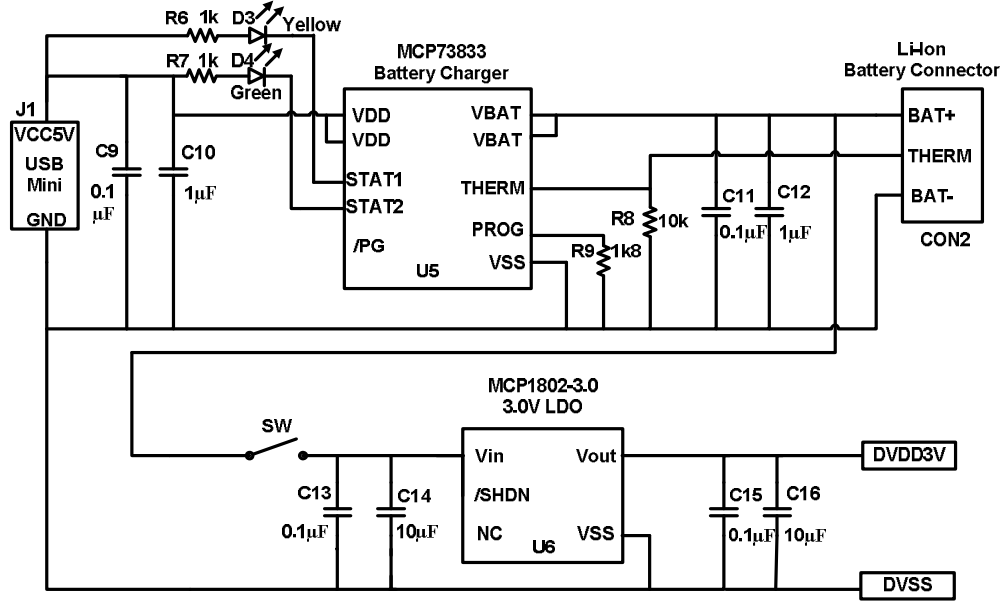


Figure 3.7: Battery charger and power supply using 3.0 V LDO.

from 5 V external dc supply. Resistor R9 (1.8 k Ω) sets the charging current as 500 mA. The R8 indicates an internal thermistor of the Li-ion battery. It is used to monitor battery temperature while charging. We are not using the temperature monitoring feature and therefore instead of thermistor, a fixed value resistor of 10 k Ω (R8) is used as suggested in datasheet [23]. The yellow LED (stat1 pin) indicates battery is charging while green LED (stat 2 pin) indicates charging is completed.

3.7 PCB design

A two-layer PCB has been designed, with both sides used to place the components to reduce the board size. The supply decoupling capacitors are placed in vicinity of power pins of the all chips. The track width of 0.5 mm has been used for supply routing (3.3 V, 5.0 V, and GND). The track width used for microcontroller and peripheral interface is 0.3 mm, except for inertial sensor, for which track width of 0.2 mm is used. The PCB size is 41 mm \times 41 mm. It has a thickness of 12 mm after soldering the components. This assembled PCB along with Li-ion battery is mounted inside an ABS plastic enclosure of dimension 14 cm \times 7 cm \times 3 cm. The PCB and placement of the FDM module in the enclosure is shown in Figure D.1 of Appendix D.

Chapter 4

HARDWARE TESTING AND SOFTWARE FOR DATA ACQUISITION

4.1 Introduction

The testing of the FDM was carried out in a stepwise manner, by writing and loading different microcontroller programs. These test programs are described in the second section. The data acquisition can be done in one of three modes: (a) sample-by-sample mode, (b) interrupt based burst-mode, and (c) polling based burst-mode. The data acquisition programs are described in the third section. The device can be controlled to start/stop data acquisition using commands through the Bluetooth transceiver. A GUI is developed using MATLAB to control various operations of the FDM and it is described in the fourth section. The magnetometer needs a calibration to be performed before using it and this calibration is described in the fifth section.

4.2 Programs for testing the hardware blocks

The following microcontroller programs have been written for testing the hardware blocks.

'LedBlink' for testing the microcontroller operation: This program initializes the microcontroller to use internal RC oscillator at 8 MHz. All ports are configured as output ports and the square wave of 0.5 Hz is generated on all the pins, which can be observed on CRO. Connecting a LED in series with 220 Ω resistor to any of the port pins should show blinking LED.

'Bluetooth_xyab for testing the wireless transceiver: The UART module of the microcontroller is configured at baud rate 115 kbps. UART receive interrupt is enabled and ISR is written to read data from the Bluetooth transceiver. If 'X' is received by UART then 'a' is transmitted and if 'Y' is received then 'b' is transmitted. Thus Bluetooth communication of microcontroller and RN42 can be verified by sensing stream of 'X' and 'Y' from the remote terminal (laptop/mobile) over Bluetooth.

'SPI_FlashMem' for testing the flash memory operation: The SPI module of the microcontroller is configured to work at 4 MHz and UART to work at 115 kbps. The global protection bits of the flash memory are reset and memory is unlocked. The manufacturer ID of the SST26 is verified. The program accepts the data bytes from the UART and stores them in memory starting from the address 0x000000. If

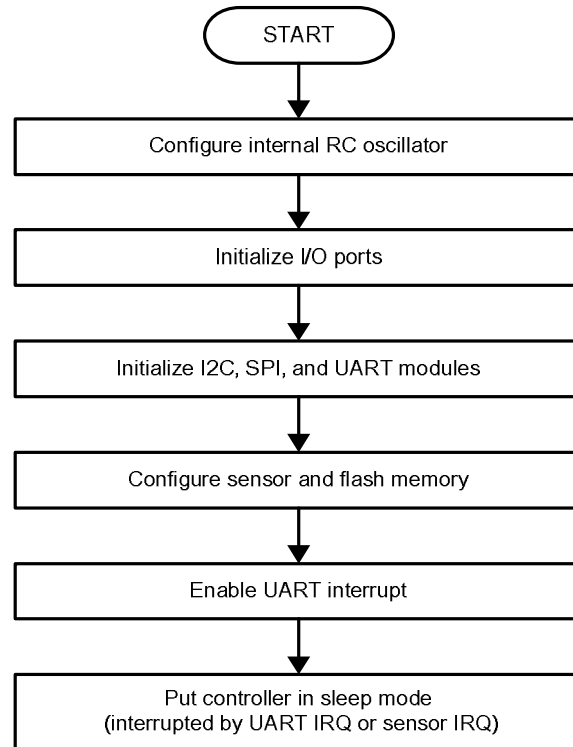


Figure 4.1 Flowchart for initial configuration setting of programs.

character ‘e’ is received, the program stops storing. It reads the previously stored data and transmits it back via UART.

'Sensor' to test the sensor: This program reads the device ID of the sensor using I2C protocol. To test the internal DAQ hardware of the sensor, the six 16-bit registers which store the values of the tri-axial accelerometer and tri-axial gyroscope are read. The sampling rate is set as 100 Hz. The baud rate is set at 115 kbps, I2C clock is set at 400 kHz and the sensor interrupt is set in sample-by-sample mode. Each time the registers get filled with data, the UART interrupt is generated. After this interrupt, the microcontroller reads the data from all the six registers using I2C protocol and transmits them through UART to Bluetooth transceiver.

4.3 Programs for data acquisition

For data acquisition, three different modes are used viz. sample-by-sample, interrupt based burst-mode, and polling based burst-mode. First microcontroller, sensor, Bluetooth and flash memory are initialized as per the requirements of the above three modes. Then the UART and sensor interrupts (INT1) are enabled, with the UART interrupt at higher priority than the sensor interrupt. The microcontroller is placed in sleep mode. The microcontroller

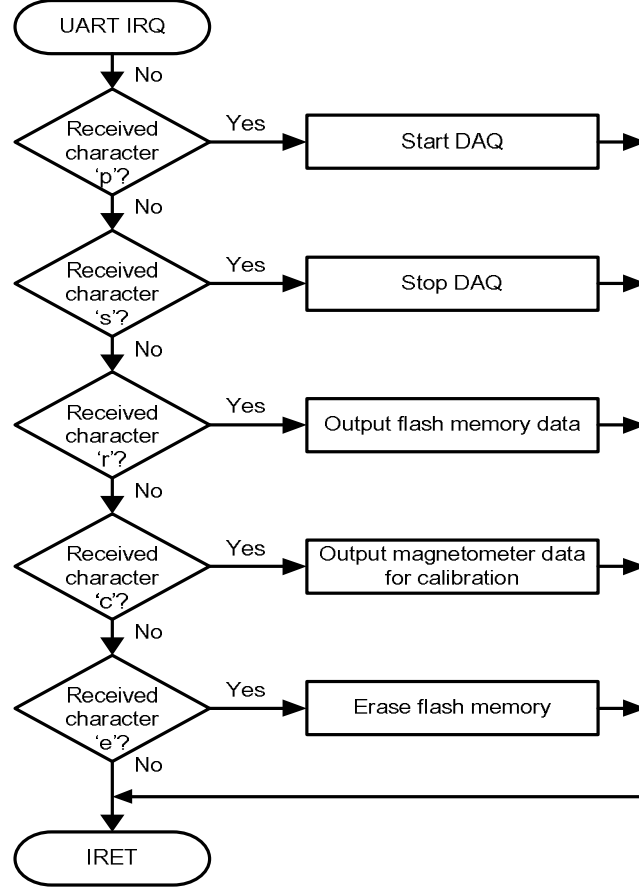


Figure 4.2: Flowchart for UART interrupt logic to control FDM operations.

can perform data acquisition using one of the three modes with help of commands provided via UART to the microcontroller over Bluetooth.

4.3.1 System initialization

The flow chart for initial configuration for microcontroller is shown in Figure 4.1. The system clock F_{osc} is set at 8 MHz and therefore internal instruction cycle clock F_{CY} ($= F_{osc}/2$) is 4 MHz. The microcontroller I/O port pins are initialized as input/output depending on their function. UART1, SPI1 and I2C2 modules of the microcontroller are enabled. The baud rate for UART1 is set to 115 kbps and the receive interrupt is enabled with highest priority. The SPI1 clock is same as the internal instruction clock F_{CY} of 4 MHz while the I2C2 clock is set at 400 kHz.

The sensor is reset initially and the full-scale range of the accelerometer is set to ± 4 g while that of the gyroscope is set to $\pm 500^\circ/s$. The DMP (digital motion processor) of the sensor chip is disabled. Sensor is configured in bypass mode so that auxiliary I2C bus is connected to the main I2C bus at 400 kHz, for direct communication with the magnetometer.

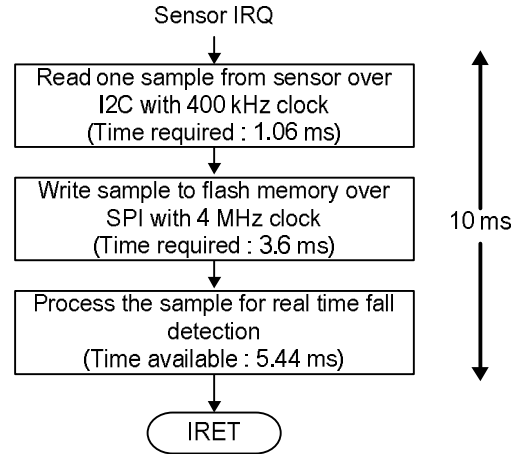


Figure 4.3: Sensor ISR in “Sample-by-sample” program.

Now magnetometer is configured for full scale range of $\pm 4800 \mu\text{T}$. The sampling frequency of accelerometer, gyroscope, and magnetometer is set at 100 Hz.

The flash memory is configured to be read in the normal mode. Use of fast mode is not required and it would consume more current. The global protection bit is reset to allow memory to be written. The UART interrupt is enabled with a priority greater than the sensor interrupt. On receiving a UART interrupt for data acquisition, the interrupt (INT1) is enabled and data are transferred from sensor to the microcontroller by servicing the INT1 ISR. To stop data acquisition, a UART interrupt is raised. It has the highest priority and hence disables the interrupt (INT1) and stops data acquisition.

The control and data transfer operations are handled through the Bluetooth transceiver which is connected to UART of the microcontroller. Single character codes are assigned for these functions. On receiving a character, UART interrupt is raised which gets the microcontroller out of sleep mode to service the ISR. A flowchart for these operations is shown in Figure 4.2. Data acquisition is started by sending ‘p’. To stop data acquisition ‘s’ is sent. The stored data is retrieved by sending ‘r’. For magnetometer calibration ‘c’ is sent. On receiving ‘c’, the microcontroller reads the current values of accelerometer, gyroscope, and magnetometer and transfers them via UART and Bluetooth transceiver.

4.3.2 Sample-by-sample mode data acquisition

In this mode, the microcontroller’s external hardware interrupt INT1 is used. Once the ‘p’ is received on the UART-Rx register, the sensor interrupt is enabled. The sensor is configured to generate ‘data ready’ interrupt at sampling rate of 100 Hz. Thus the INT1 ISR executes every 10 msec. Figure 4.3 shows the flowchart of the ISR for sample-by-sample data acquisition mode. The 18 bytes of data are read, i.e., 6 bytes each for accelerometer, gyroscope and magnetometer. Microcontroller can come out of this continuous acquisition

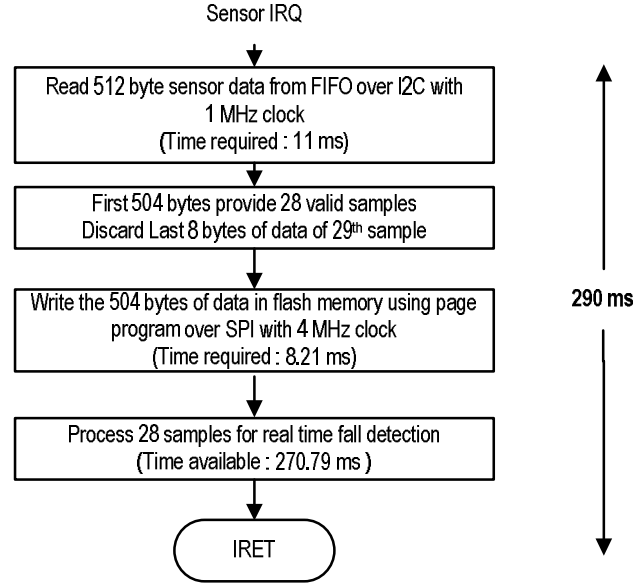


Figure 4.4: Sensor ISR in “Burst-mode interrupt” program.

process only by a UART IRQ followed by receipt of ‘s’ to stop data acquisition. In this mode, the sensor is configured in the bypass mode so as to access magnetometer registers and hence I2C clock frequency configured as 400 kHz. The 18 bytes are then transferred to the flash memory over SPI bus. The time measured using internal timer module, to read one sample (18 bytes) from sensor over I2C bus at 400 k Hz is 1.06 ms and the time measured to store one sample over SPI bus at 1 MHz is 3.6 ms. Thus total time required to read and store one sample is 4.66 ms. This gives 5.44 ms time to process the sample to determine the fall condition.

4.3.3 Interrupt based burst-mode data acquisition

The time required to read samples from sensor and write them to flash can be reduced by using sensor FIFO. The sensor has 512 byte FIFO which can store the 28 samples ($18 \times 28 = 504$). The FIFO overflows every 29 samples. An interrupt is generated on FIFO overflow, which is used to read data in burst-mode at 1 MHz over same I2C master bus. The bypass mode is disabled and sensor’s internal master I2C module is used to read magnetometer data. The master I2C module reads 6 bytes corresponding to three axes of magnetometer to external sensor data registers (EXT_SENS_DATA_00- EXT_SENS_DATA_05). The sensor is configured to store the 12 bytes from accelerometer and gyroscope along with 6 bytes from external sensor registers to the FIFO at sampling rate of 100 Hz. Thus at every 290 ms an interrupt is generated and corresponding ISR (INT1_ISR) is executed. The flowchart for interrupt based burst-mode ISR is shown in Figure 4.4. As data are read only from master I2C bus, its clock is configured at 1 MHz which provides faster data transfer as compared to

sample-by-sample mode. The time to read sensor FIFO (512 bytes) as measured using internal timer of PIC is 11 ms. This leads to loss of one sample during FIFO read operation, as time required to read is more than the sample duration (10 ms) and FIFO cannot be written to, unless previous data is read by the microcontroller. Further time required to store the 28 samples (504 bytes) in flash memory is measured as 8.21 ms. Thus time required to read data from the sensor and store them to the flash memory is 19.21 ms (11 ms + 8.21 ms), leaving 270.79 ms (290 ms – 19.21 ms) for processing 29 samples. This increases the time available for processing each sample from 5.44 ms to 9.33 ms. In this mode, the last sample (29th sample) is not recorded completely as FIFO is not exact multiple of 18 bytes. Every time the FIFO is written to, the 29th sample is recorded partially and hence this sample is discarded. Thus in the interrupt based burst-mode, at least two samples are lost during data read operation. While storing the data to flash memory, the page program method is used. In page program method, memory is divided in 256-byte pages. It does not allow writing data at any arbitrary address. The new block of data has to be written at page boundaries. To write 504 bytes to flash memory, first this block is divided in two blocks of 252 bytes and then these two blocks are written to two consecutive memory pages. Thus 4 bytes from every page of 256 byte remain unused, i.e., wasting about 1.5% of memory space.

4.3.4 *Polling based burst-mode data acquisition*

This mode is similar to the interrupt based mode except microcontroller does not wait for FIFO overflow. The sensor has FIFO count register which keep track of how many bytes are written to FIFO during data acquisition. In polling mode FIFO count register is continuously read by the microcontroller until 450 bytes are acquired. The count of 450 corresponds to 25 samples where each sample constitutes of 18 bytes. Further it takes 9.74 ms to read 450 bytes from FIFO, which is less than sampling interval of 10 ms and hence no sample is lost. These 450 bytes are then written to the flash memory which takes 7.47 ms. Thus out of 250 ms time which is available for reading, storing, and processing the 25 samples, 17.21 ms is required to read and store the data. Thus time available for processing is 232.79 ms, which gives 9.31 ms for processing one sample, same as in the interrupt based data acquisition. The flowchart for polling based data acquisition is shown in Figure 4.5. The block of 450 byte is divided into two blocks of 225 bytes and then stored in two continuous pages using page program. In this case, 31 bytes of memory space is not used per page, i.e., wastage of about 12% of memory space.

4.4 **GUI for data acquisition**

The graphical user interface (GUI) is developed using MATLAB for offline processing of the data acquired from FDM. Figure 4.6 shows the screenshot of GUI. When

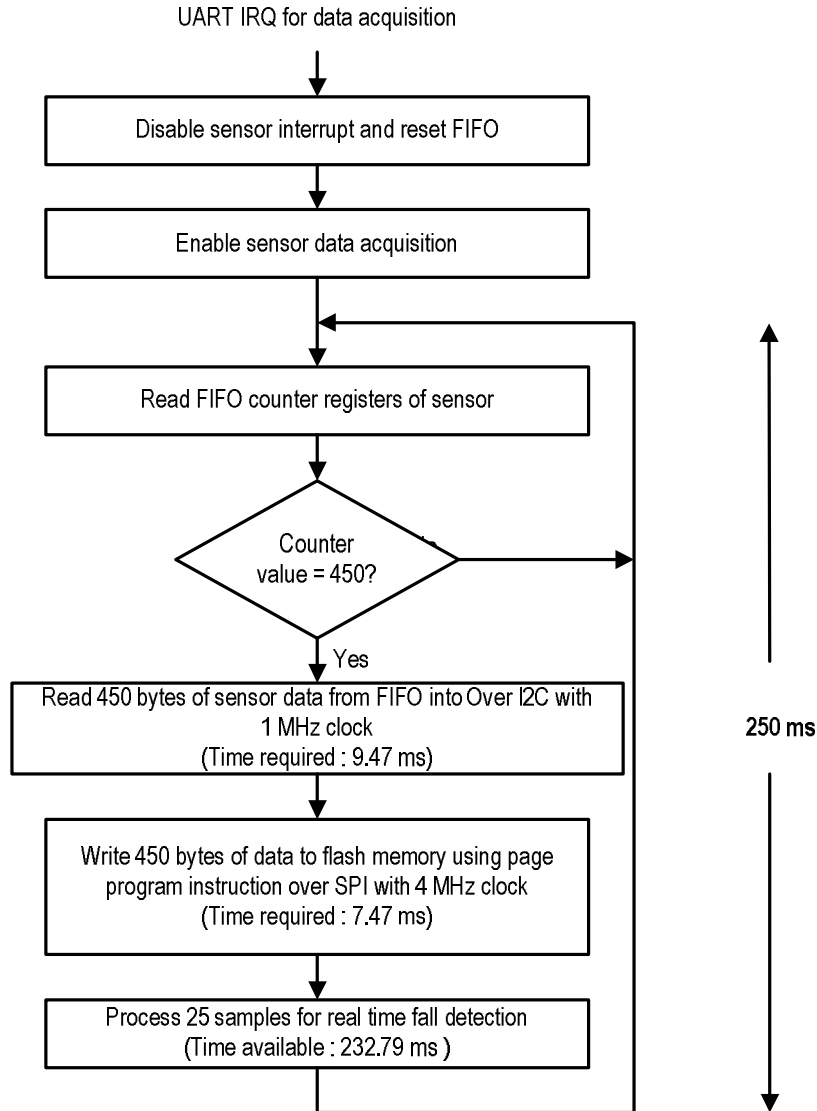


Figure 4.5: Data acquisition in “Burst-mode polling” program.

CONNECT button is pressed, the ‘COM4’ port is opened for serial communication using the Bluetooth transceiver. The COM-port is opened and configured to communicate serially at 115 kbps. Now character ‘A’ is sent over Bluetooth to detect FDM, which is programmed to send character ‘a’ on receipt of ‘A’. On pressing DISCONNECT button the COM4-port is closed. Further, predefined characters ‘p’, ‘s’, ‘r’, ‘e’, and ‘c’ are sent over computers Bluetooth to control various functions of FDM, on clicking respective buttons viz. START DAQ, STOP DAQ, RETRIEVE DATA, ERASE MEMORY, and CALIBRATE. On pressing STAR DAQ ‘p’ is sent to FDM and FDM starts data acquisition in one of the three modes viz. sample-by-sample, interrupt based burst-mode, or polling based burst-mode.

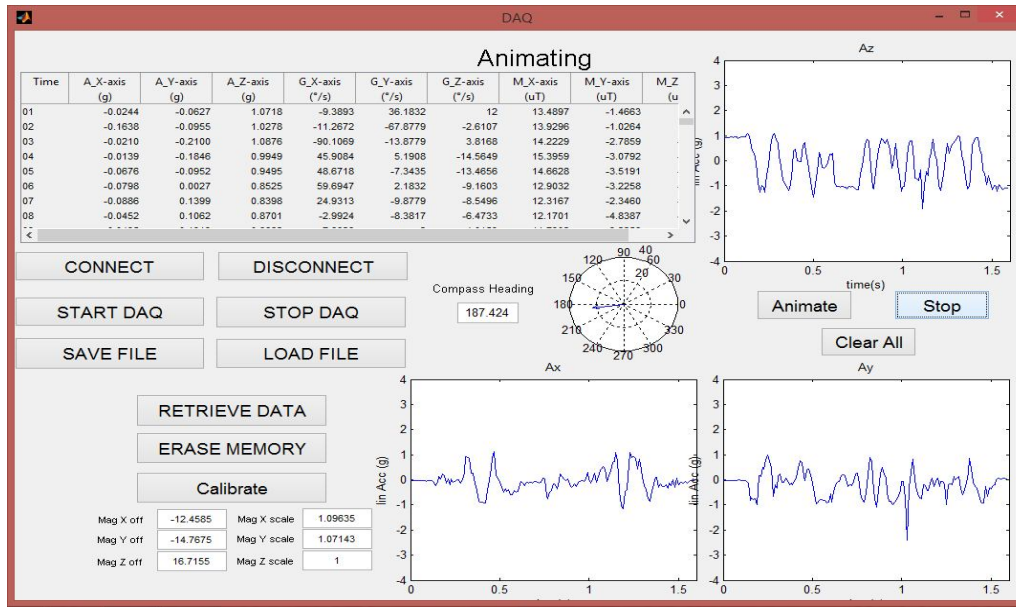


Figure 4.6: GUI to control FDM.

If a fall is detected, GUI receives character 'FALL' and indication for fall is prompted in the message box. The data acquisition is stopped by pressing STOP DAQ button which sends character 's'. The data acquired in the flash memory is read from FDM using RETRIEVE button which sends 'r' as retrieve command. Once data is retrieved then it can be stored to the computer in '.dat' format using SAVE FILE button. This data can be further processed offline. The LOAD FILE button is used to read data stored in '.dat' file and display variations in various signals corresponding to accelerometer, gyroscope, and magnetometer graphically. The real-time variations in accelerometer, gyroscope, and magnetometer signals are displayed by pressing ANIMATE button. It is found that magnetometer readings are affected by nearby metallic parts and hence need calibration. This calibration can be done using CALIBRATE button. The FDM needed to be rotated in various orientations on pressing CALIBRATE button for next three minutes. In this mode magnetometer data along all three axes are recorded and required calibration factors are calculated as discussed in the next section.

4.5 Magnetometer calibration and testing

Magnetometer is used to measure the magnetic field, to find the sensor orientation with reference to the local magnetic field of the earth. As the earth's magnetic field gets distorted due to the presence of magnets, soft iron, and current carrying conductors, different types of errors get added to the magnetic field measurement. Plots of the magnetic field measured along the X and Y axes, for different rotations of the module are shown in Figure 4.7. The device orientation or heading in degrees is calculated by using the following relation.

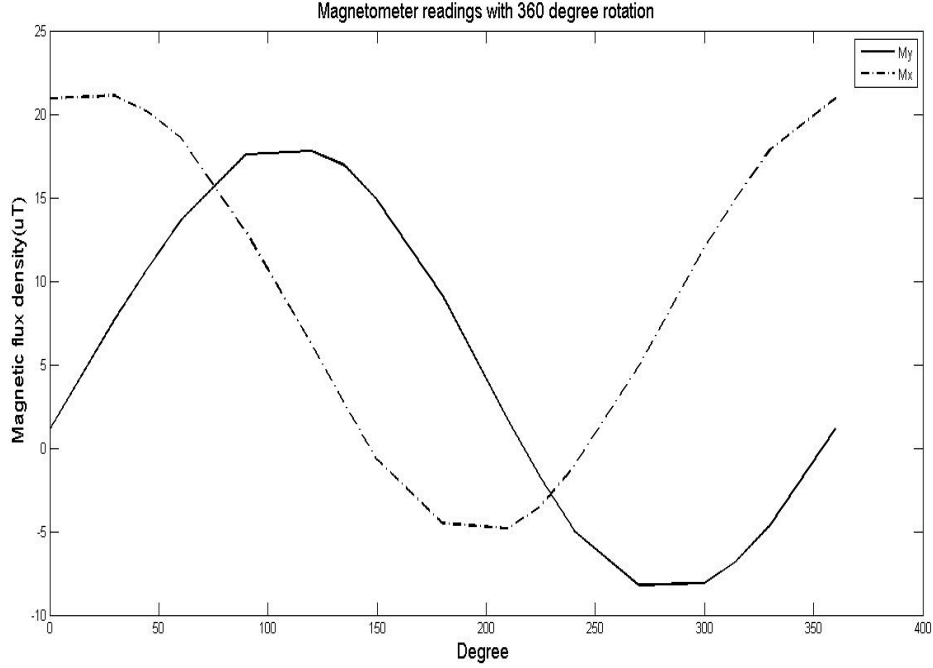


Figure 4.7: Magnetometer reading along X and Y axis for 360° rotation of FDM.

if ($y > 0$), compass heading = $90 - (180 / \pi) \tan^{-1} (x / y)$

if ($y < 0$), compass heading = $270 - (180 / \pi) \tan^{-1} (x / y)$

if ($y = 0, x < 0$), compass heading = 180

if ($y = 0, x > 0$), compass heading = 0

It can be seen from the graph that the plots are not symmetric, that is, maximum and minimum values measured along x and y directions are not identical. This is because of the existence of stray magnetic field which cannot be avoided. The magnetometer readings get affected by existence of nearby metallic parts including the mounting screws. Also it is seen that in indoor environment, there always exist stray magnetic fields generated by nearby equipment and metal used in construction of the building. The change in the magnetic field is measured by moving device laterally without changing its orientation. During the readings, care was taken that device was not in the vicinity of any metallic objects and was well elevated from the ground surface. There still were deviations of about 0.5-1 μT . To reduce the effect of stray magnetic field and to make response along x y z axis symmetrical, offset and scaling may be applied to the raw magnetometer readings. Song et al. [16] have suggested the method below to find the offset and scaling factor to be applied to raw data.

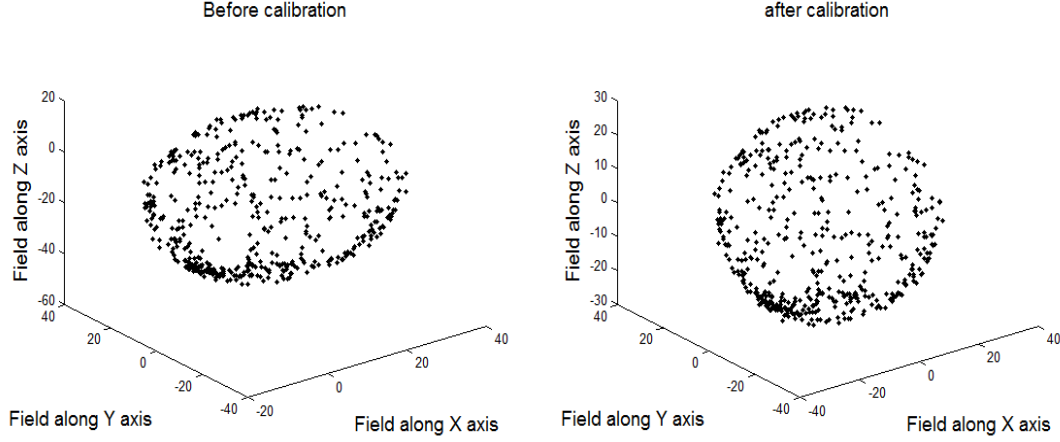


Figure 4.8: Magnetometer data before and after calibration.

$$X_{sf} = \max(1, (Y_{max} - Y_{min}) / (X_{max} - X_{min})) \quad (4.1)$$

$$Y_{sf} = \max(1, (X_{max} - X_{min}) / (Y_{max} - Y_{min})) \quad (4.2)$$

$$X_{off} = ((X_{max} - X_{min}) / 2 - X_{max}) X_{sf} \quad (4.3)$$

$$Y_{off} = ((Y_{max} - Y_{min}) / 2 - Y_{max}) X_{sf} \quad (4.4)$$

$$X_{new} = X_{sf} X_{raw} + X_{off} \quad (4.5)$$

$$Y_{new} = Y_{sf} Y_{raw} + Y_{off} \quad (4.6)$$

Using this method, the scaling factor and offset values along all three axes are found. These values are used as calibration values for the magnetometer. Figure 4.8 shows the magnetometer data before calibration and after calibration. After calibration plot is seems to be more symmetric about the origin. The magnetometer is further tested for linearity and saturation. The experiment was performed by moving bar magnets towards the magnetometer axis and magnetic field strength is measured. This experiment was repeated by adding different number of magnets in series to increase magnetic field. Figure 4.9 shows the graphs plotted for magnetometer readings with different number of magnets placed parallel and anti-parallel to the magnetometer x-axis.

The results show that we may need fresh calibration of the magnetometer at each location of use and hence it may not be useful as sensor for fall detection in the wearable device due to change in the location of the person. However, it may be very useful for actigraphy.

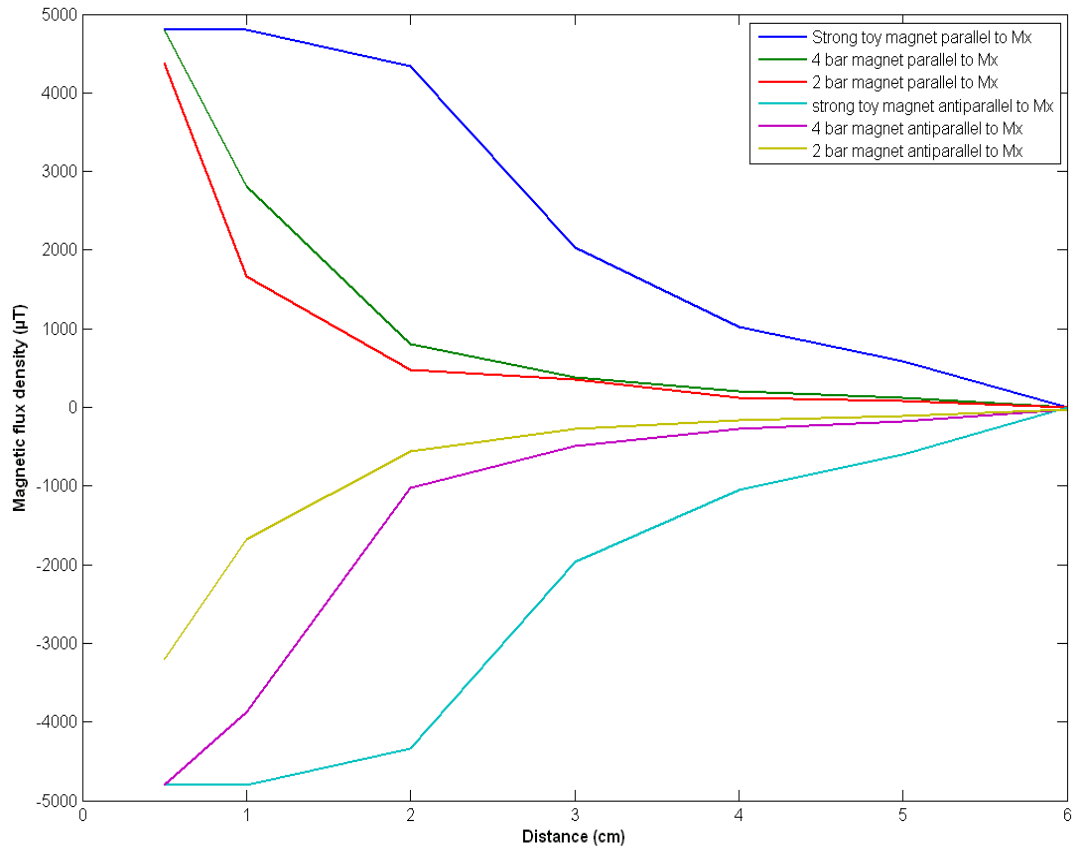


Figure 4.9: Magnetometer readings for different magnets.

Chapter 5

SIGNAL PROCESSING FOR REAL-TIME FALL DETECTION

5.1 Introduction

To investigate signal processing techniques for discriminating between ADL and fall, signals are acquired with the device subjected to movements related to ADL and simulated fall. The techniques for these investigations are implemented using MATLAB, with consideration for their suitability for real-time processing. Subsequently, the finalized technique is implemented for real-time processing using the microcontroller of the device and tested. The subsequent sections present the test setups for signal acquisition, investigations for signal processing, implementation for real-time processing, and test results.

5.2 Test setups for signal acquisition

For investigating and developing an appropriate technique for fall detection, the sensor data were recorded for various activities of daily life (ADL) and simulated fall. The device was worn and 8 activities of daily life, viz. (i) jumping, (ii) walking, (iii) running, (iv) jogging, (v) hopping, (vi) sitting down on a chair and getting up, (vii) sitting down on a chair and getting up with jerk, and (viii) walking up and down the staircase, were carried out by 4 volunteers (age: 23 – 28 years, weight: 70 – 100 kg, height: 161 – 180 cm). No fall related recordings were taken from the volunteers to avoid possible injury to them. The fall conditions were simulated by tying the device to the end of a stick of about waist height (91.4 cm) as shown in Figure 5.1. Signals were recorded for a total of 16 falls of the stick, with the FDM facing towards north, south, east, and west and making the stick fall in front, back, left, and right side for each of above four directions.



Figure 5.1: FDM fixed at top of stick to record simulated falls.

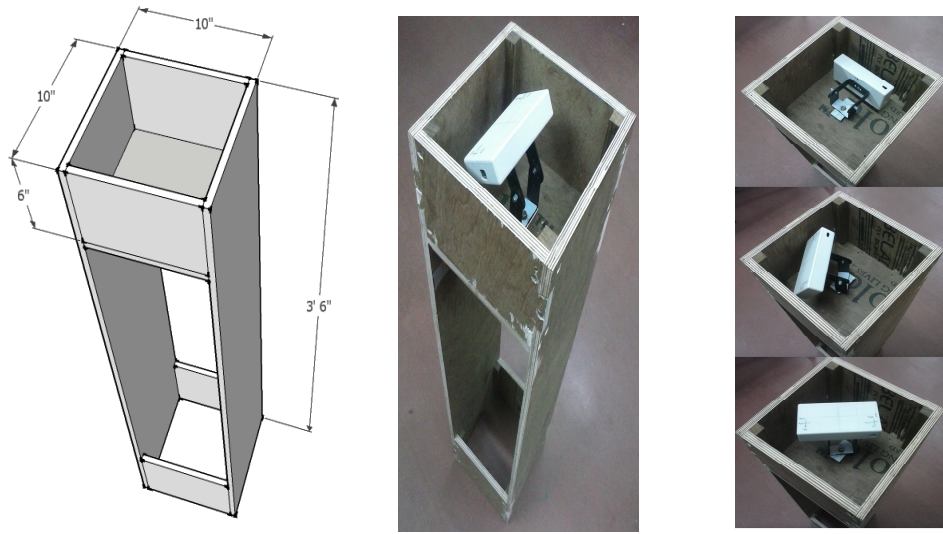


Figure 5.2: Wooden box fabricated for simulated fall with different orientations of FDM.

Further, to investigate the effect of device orientation on fall detection, an experimental setup was designed for recording of the fall with different orientations of the device with reference to the torso. In this setup, a wooden box of height of 106.6 cm (3'6") and cross-section of 25.4 cm \times 25.4 cm (10" \times 10") was used to approximate the human body up to waist height, as shown in Figure 5.2. The box was provided with a wooden base plate at the height of 91.4 cm (3'), and supporting brackets were installed on this plate to hold the device. These brackets, also shown in Figure 5.2, provided two degrees of freedom. The device could be rotated in two orthogonal planes, horizontal (along yaw axis) and vertical (along pitch axis). Using this setup, simulated falls with different orientations of the device were recorded. The data were recorded by keeping the device fixed along five pitch orientations of 0°, 30°, 45°, 60°, and 90°. For every pitch orientation, FDM is rotated along the yaw axis at 16 angles over 0° – 330°. Thus, data were recorded for the box falling with 80 different orientations of the device.

5.3 Investigations for signal processing

The fall detection device developed by Kumar [28] used only accelerometer to detect the fall. In the current device, use of three sensors viz. accelerometer, gyroscope and magnetometer is considered in order to improve discrimination between fall and ADL.

In earlier reported fall detectors using threshold based discrimination, the three axial components of the acceleration or the acceleration magnitude is used. In the fall detector reported by Kumar and Pandey [7] it was proposed that decomposition of the acceleration vector along multiple directions can be used for improving the fall detection and making the

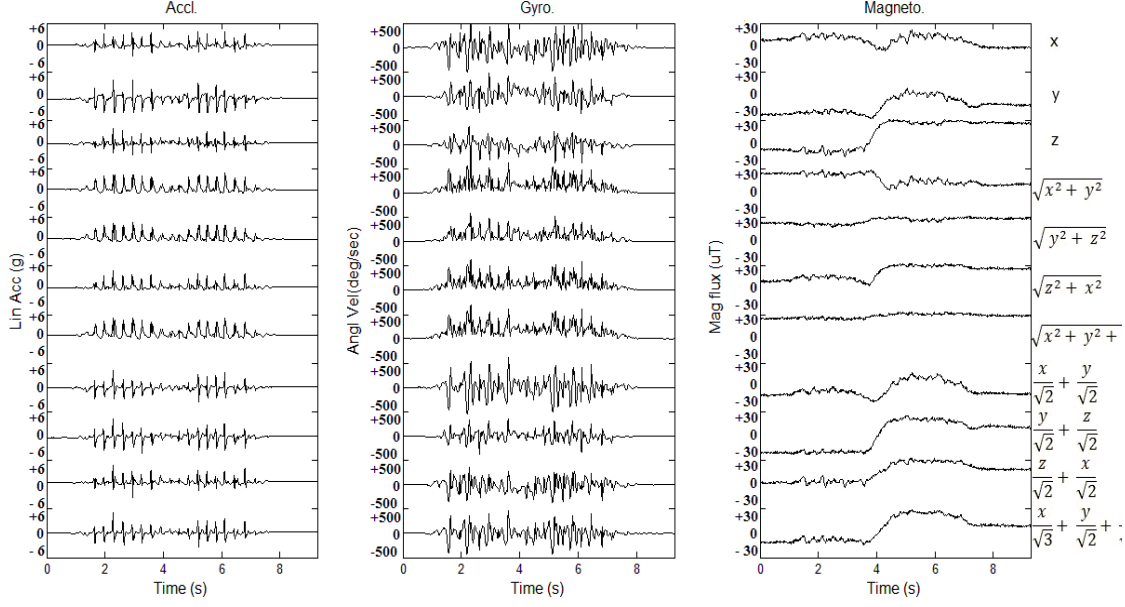


Figure 5.3: Sensor outputs and corresponding decompositions for the running activity.

detection performance independent of the orientation of the device with respect to the human torso. Taking this approach forward, we plotted components of the sensor outputs along eleven directions. From the x , y , and z components of the vector, the vector magnitude and the magnitudes in the three plains are calculated as the following:

$$m_{xyz} = \sqrt{x^2 + y^2 + z^2} \quad (5.1)$$

$$m_{xy} = \sqrt{x^2 + y^2} \quad (5.2)$$

$$m_{yz} = \sqrt{y^2 + z^2} \quad (5.3)$$

$$m_{zx} = \sqrt{z^2 + x^2} \quad (5.4)$$

In addition, the components of the signal vector along the diagonal of the xy , yz , and zx plains are calculated as the following:

$$c_{xy} = (x + y)/\sqrt{2} \quad (5.6)$$

$$c_{yz} = (y + z)/\sqrt{2} \quad (5.7)$$

$$c_{zx} = (z + x)/\sqrt{2} \quad (5.8)$$

Component of the signal vector along the direction with equal angle with the three axes is calculated as:

$$c_{xyz} = (x + y + z)/\sqrt{3} \quad (5.9)$$

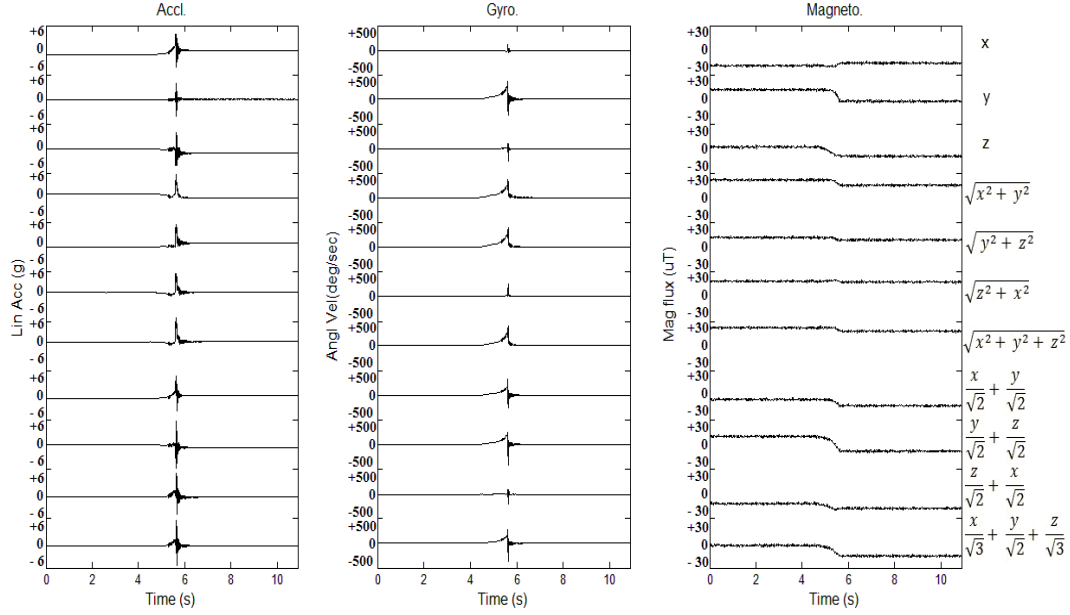


Figure 5.4: Sensor outputs and corresponding decompositions for simulated fall with device orientation of yaw 0° and pitch 0° .

The set of 11 variables consists of components along seven directions (x , y , z , c_{xy} , c_{yz} , c_{zx} , c_{xyz}) and four magnitudes (m_{xyz} , m_{xy} , m_{yz} , m_{zx}). It is assumed that a fall will be captured as a large deviation in some of these variables, providing a computationally inexpensive and robust technique to detect the fall.

An example of the plots of the 11 variables for ADL of running is shown in Figure 5.3 and that for a simulated fall is shown in Figure 5.4. Several other examples of the plots are provided in Appendix F for ADL and in Appendix G for fall. These plots show that fall is associated with much larger deviation from the baseline along some of the directions as compared to the deviations in case of ADL.

5.3.1 Deviation detector for discrimination of ADL and fall

After analyzing the fall and ADL data, it is seen that fall is associated with rapid change in the signal amplitude, as compared to ADL. Further, the baseline of the signal varies with device orientation and decomposition direction. Hence to discriminate between fall and ADL, a change from the baseline needs to be detected.

The proposed signal processing for deviation envelope detection is shown in Figure 5.5. The first step consists of baseline estimation. The baseline can be estimated by moving average filter or some other low-pass filter. However, a moving median filter is considered to be more effective as it rejects the deviations based on their duration and the output is not affected by amplitude of the deviation. Hence a moving median filter is used as the baseline

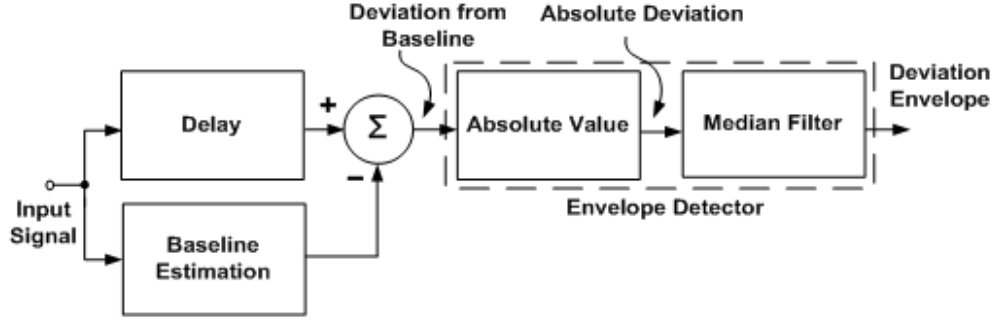


Figure 5.5: Signal processing for detection of deviation envelope of signal component.

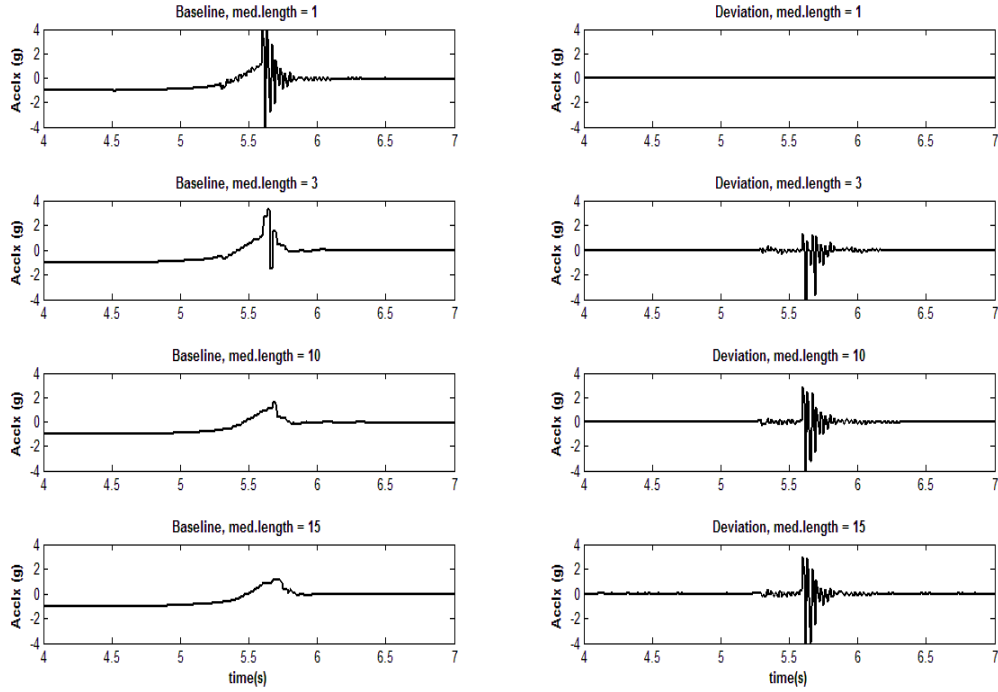


Figure 5.6: Effect of different lengths of median filter on baseline and deviation from baseline, for the component a_x in Figure 5.4.

estimator. The deviation from the baseline is calculated by subtracting the baseline from the signal. As the median filter results in a delay approximately equal to half of the filter length, the signal is correspondingly delayed before taking the difference. Figure 5.6 shows the effect of different lengths of median filter and corresponding baseline deviation. To detect the envelope of the deviation, the absolute values is low-pass filtered using a median filter. After analyzing the signals with sampling frequency of 100 Hz for fall and ADL, the length of median filter for the baseline estimation is selected as 15 samples and that for the envelope detection is selected as 5 samples. An example of results of the signal processing at various steps is shown in Figure 5.7.

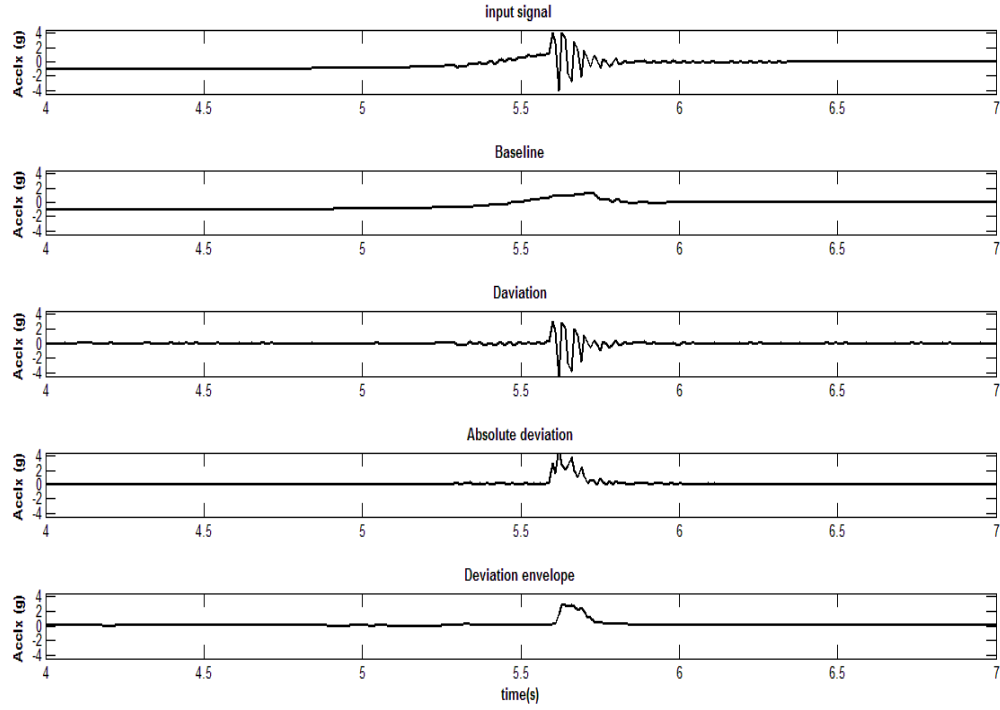


Figure 5.7: Example of signal processing results: input signal, estimated baseline, deviation from baseline, absolute deviation, deviation envelope, for the component a_x in Figure 5.4.

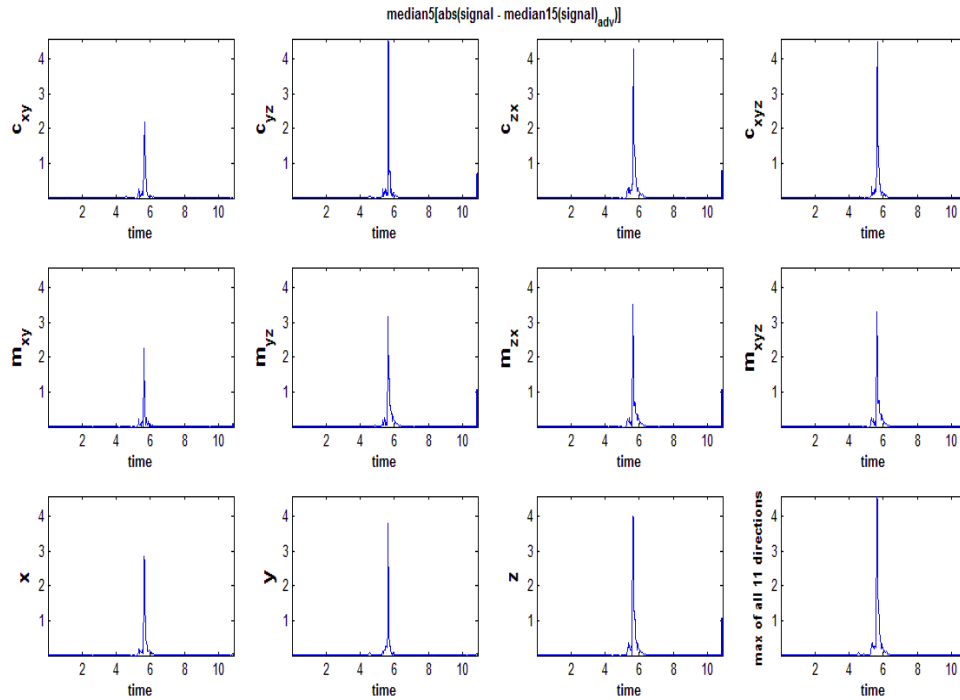


Figure 5.8: Deviation envelopes of the 11 decompositions of the accelerometer output and the corresponding maximum, for the fall data as in Figure 5.4.

Table 5.1: Variables which best capture fall out of $x, y, z, c_{xy}, c_{yz}, c_{zx}, c_{xyz}, m_{xyz}, m_{xy}, m_{yz}, m_{zx}$.

	Pitch 0°	Pitch 30°	Pitch 45°	Pitch 60°	Pitch 90°
Yaw 0°	c_{yz}, c_{xyz}	c_{zx}	c_{xyz}	c_{zx}, c_{xyz}	c_{yz}
Yaw 30°	c_{yz}	c_{yz}	x, z	c_{xyz}	c_{yz}
Yaw 45°	y, z, c_{zx}	x, c_{yz}	y	c_{yz}	c_{yz}
Yaw 60°	c_{yz}	c_{yz}	y	c_{yz}	c_{yz}
Yaw 90°	c_{yz}	y	y	z	c_{xy}
Yaw 120°	c_{yz}	x, y, z	c_{xy}	y	y
Yaw 135°	y	c_{yz}	y	c_{xyz}	c_{xy}
Yaw 150°	m_{xyz}	c_{xy}, y	c_{yz}	c_{zx}	c_{yz}
Yaw 180°	z	c_{xyz}	c_{yz}	c_{zx}	x, z
Yaw 210°	c_{xy}, c_{yz}	y, c_{zx}	c_{zx}	c_{yz}	c_{yz}, z
Yaw 225°	c_{xyz}	y	c_{yz}	y	c_{yz}
Yaw 240°	c_{yz}	c_{yz}	c_{xyz}	y	y
Yaw 270°	y	y	y	y	y
Yaw 300°	z	c_{zx}	c_{xy}	y	c_{zx}
Yaw 315°	z	c_{yz}	c_{zx}	c_{xyz}	y
Yaw 330°	c_{xy}	x	x, c_{zx}	c_{yz}	c_{yz}

5.3.2 Testing effectiveness of multidimensional decomposition for fall detection

From the recorded raw data of acceleration along the three axes, eight more variables ($c_{xy}, c_{yz}, c_{zx}, c_{xyz}, m_{xyz}, m_{xy}, m_{yz}, m_{zx}$) as described earlier were generated and processed for obtaining the deviation envelope as described in Figure 5.5. The results of this processing for accelerometer signals are shown in Figure 5.8 for device orientation with yaw = 0° and pitch = 0°. In this figure, a twelfth plot is added showing the instantaneous maximum value of the 11 variables. It is seen that the fall is best captured by the variables c_{yz} and c_{xyz} . The results for fall with all 80 orientations were analyzed and it was found that different sets of variables captured the fall in different cases, as summarized in Table 5.1. Thus, it is found that decomposing the signal along these 11 directions and processing the components is useful for orientation-independent capturing of the fall. Further, taking the maximum of the 11 variables is a computationally simple and robust indicator of the fall.

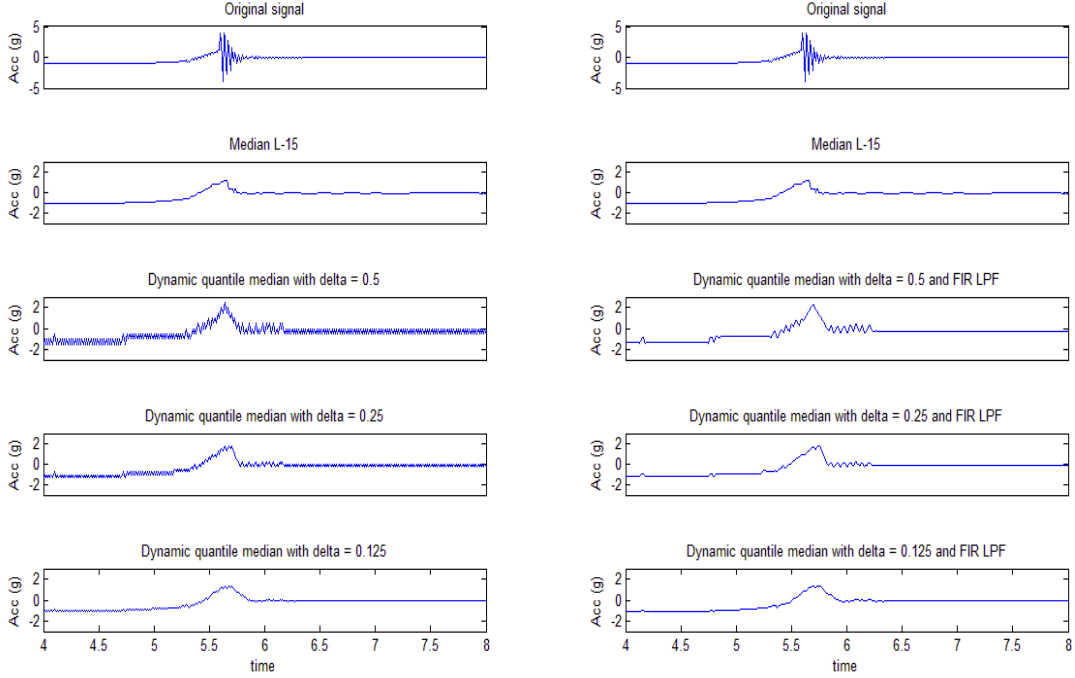


Figure 5.9: Comparison of true median and median with dynamic quantile tracking, for the fall data in Figure 5.4.

5.3.3 Median filter using dynamic quantile tracking

The median filter requires sorting of the input samples, which is time consuming. It was found that the time required to find real-time median of length 15 (for baseline estimation) and median of length 5 (for deviation envelope) for all the 11 variables is more than 10 ms with PIC microcontroller with system clock at 4 MHz. The dynamic quantile tracking as proposed by Tiwari and Pandey [17] can be used for significantly reducing the computation in median tracking. In this technique, the p -quantile $D(n)$ of the input sequence $x(n)$ is estimated by incrementing or decrementing the previous estimate, by appropriate fraction of the input signal range R as the following:

$$D(n) = \begin{cases} D(n-1) + \Delta_+, & x(n) \geq D(n-1) \\ D(n-1) - \Delta_-, & x(n) < D(n-1) \end{cases} \quad (5.10)$$

$$\Delta_+ = \lambda p R \quad (5.11)$$

$$\Delta_- = \lambda (1 - p) R \quad (5.12)$$

Where λ is the step size control factor, and $1/\lambda$ approximately corresponds to the quantile estimation length. A small value of λ results in low ripple but slow convergence of the estimation. In the method described in [17], the range is dynamically estimated by using recursive relations for peak and valley estimation. In our application, the range R of the signal can be taken as the sensor range and does not have to be estimated. For accelerometer, we

take $R = 8g$. For median estimation, $p = 0.5$. To remove the ripple in the output, a 6-point low-pass FIR filter is used with difference equation as the following:

$$y(n) = -0.0805x(n) + 0.1966x(n-1) + 0.4776x(n-2) + 0.4776x(n-3) + 0.1966x(n-4) - 0.0805x(n-5) \quad (5.13)$$

Figure 5.9 shows a comparison of true median and median estimated by dynamic quantile tracking. Based on these plots, it is inferred that 15-point moving median can be approximated by dynamic quantile tracking with $p = 0.5$ and $\lambda = 1/8$.

5.3.4 Median filter using quick sort for true median

It was observed that dynamic quintile tracking with fixed assumed range does not provide satisfactory results if there is a large variation in the range, as in case of the 5-point median filter used for envelope detection. Therefore, use of quick sort based median filter was investigated as it is faster than the traditional bubble sort based median filter. The flowchart of quick sort is shown in Figure 5.10. In this method, three arrays of length N are required to find N -point median. The first array is used as FIFO to store the past the N input samples and two more arrays `buf_0` and `buf_1` are used for sorting. The algorithm assumes that one of the buffers holds the last N samples, which are already sorted in ascending order. To find moving median corresponding to the new sample $x(n)$, the oldest sample $x(n-N)$ is to be discarded and buffer is to be shifted, while inserting current value $x(n)$ at appropriate location in ascending order. Instead of shifting the buffer we copy its contents to another buffer, while discarding the oldest sample and inserting the new sample at the appropriate location. The algorithm starts with selecting `buf_0` as source and `buf_1` as destination, where `buf_0` is assumed to be initially sorted. The contents of `buf_0` are compared with $x(n)$ and $x(n-N)$ while copying them to `buf_1`. If $x(n)$ is greater than the current value of `buf_0`, then the current value of `buf_0` is copied to `buf_1`, otherwise $x(n)$ is copied to `buf_1`. If the current value of `buf_0` is equal to $x(n-N)$, then this value in `buf_0` is discarded. For subsequent input value, the buffer functions are swapped, i.e. `buf_1` becomes source and `buf_0` becomes destination, by toggling the `buf_sel` flag. After being copied to the destination buffer, $x(n)$ is over-written with a number outside the range of input signal, to avoid its multiple copies. The number of comparisons required to sort array of length N , for traditional bubble sort is $N(N-1)/2$, while that required for quick sort is N . The processing delays in the algorithms was tested for implementation using the microcontroller to find 5-point moving median of input sampling frequency of 100 Hz. The execution time for real-time processing was found to be 3.14 ms with quick sort and 6.30 ms with bubble sort. Therefore, there is a scope for using quick sort for 5-point moving median although not for 15-point moving median.

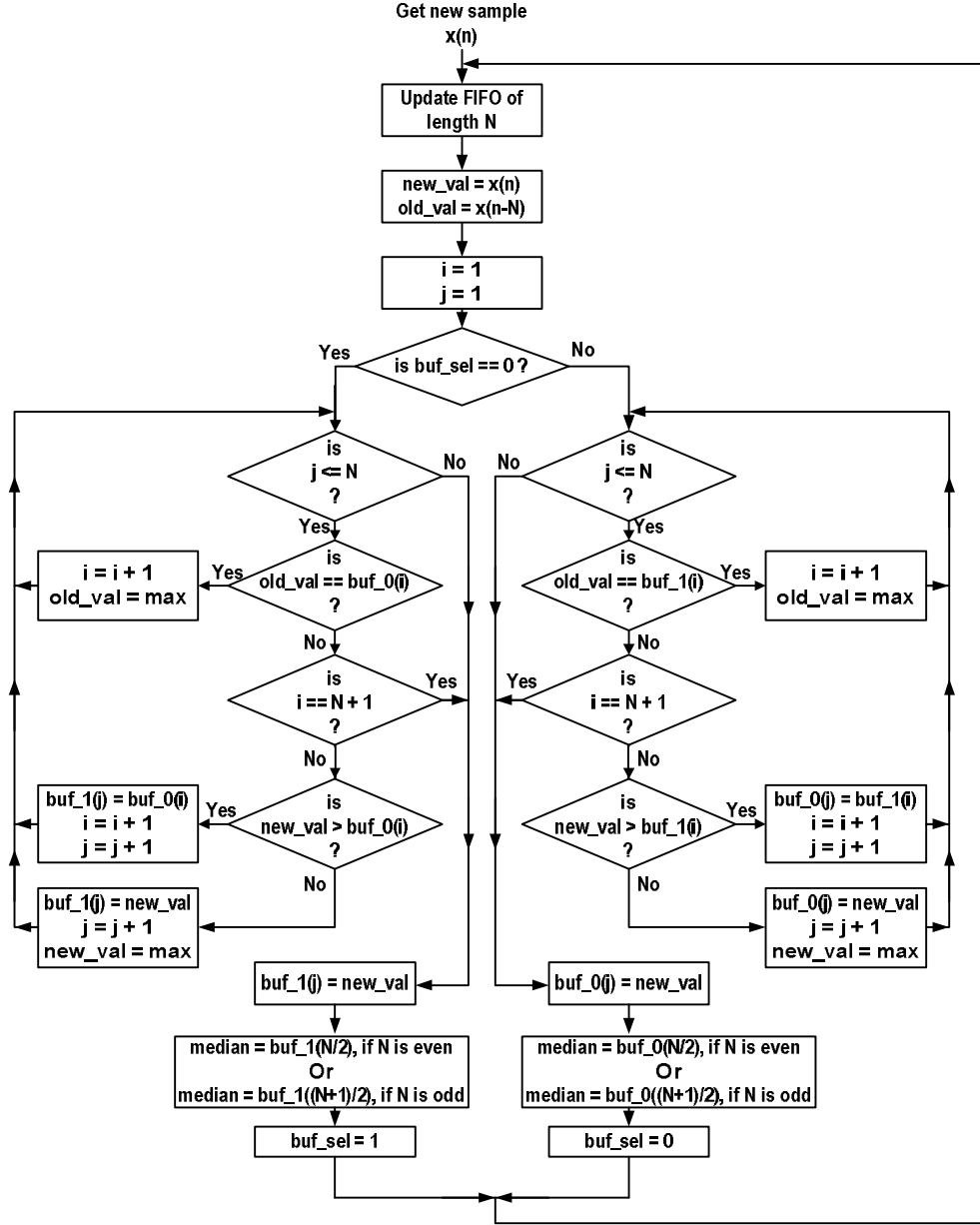


Figure 5.10: Flowchart for real-time median filter using quick sort.

5.3.5 Squaring of input signal for contrast enhancement

To further improve the contrast between ADL and fall, the input signal was squared before processing for baseline and deviation. Figure 5.11 shows an example of results of signal processing with squaring of the input signal. For this comparison, results of two ADL activities and a fall with yaw = 0° and pitch = 0° are considered. The instantaneous maximum values out of all 11 variables (x , y , z , c_{xy} , c_{yz} , c_{zx} , c_{xyz} , m_{xyz} , m_{xy} , m_{yz} , m_{zx}).

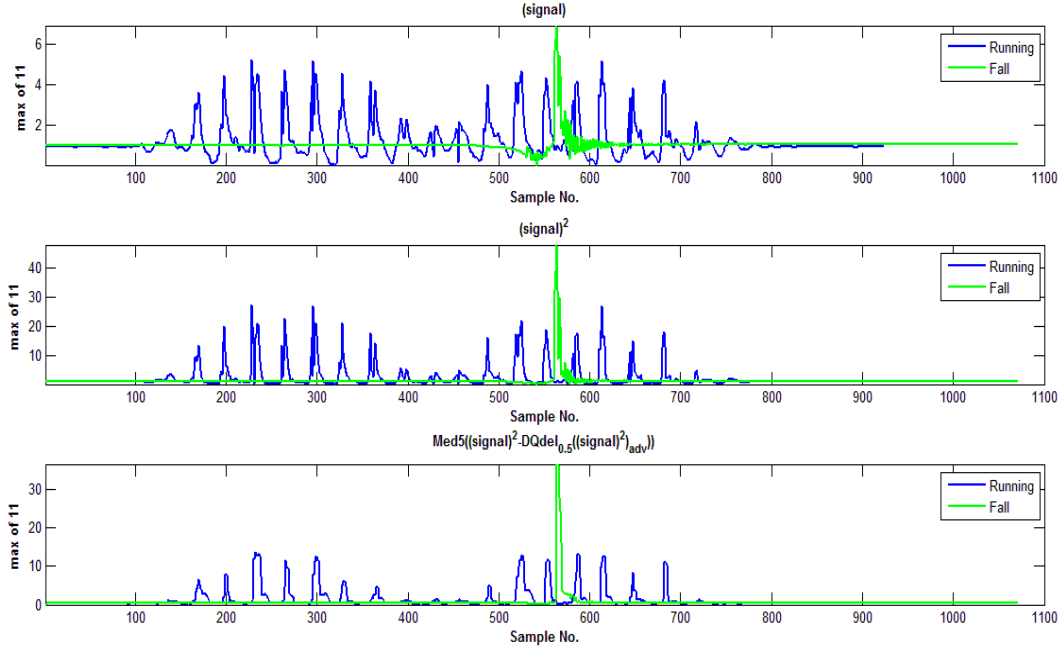


Figure 5.11: Example of signal processing results with squaring of the input signals for running and fall as in Figures 5.3 and 5.4, respectively.

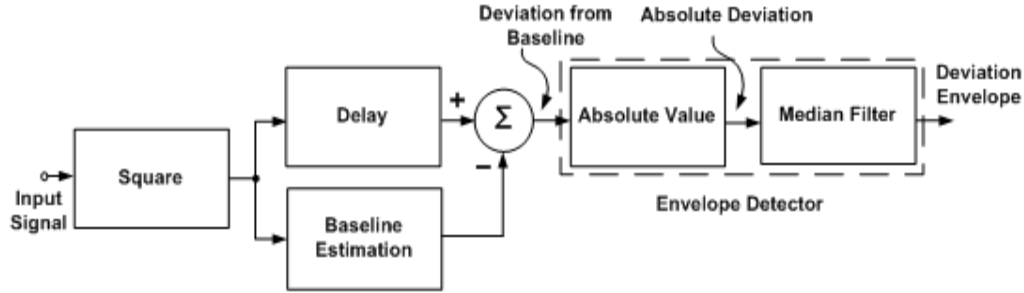


Figure 5.12: Revised signal processing for detection of deviation envelope.

5.4 Implementation for real-time processing

Based on the investigations reported in the earlier section, the signal processing for real-time detection of deviation envelope was revised and is shown in Figure 5.12. The input signal is squared and is used for baseline estimation using an approximation of 15-point median filter realized by dynamic quantile tracking with fixed range $R = 8$, $p = 0.5$, and $\lambda = 1/8$. The moving median is followed by 6-point low-pass FIR filter for ripple suppression. It was found that a delay of 3 samples in the signal path before deviation calculation was adequate. The deviation envelope estimation involves 5-point median implemented using quick sort. This deviation envelope detection was implemented for each of the eleven variables obtained from the tri-axial accelerometer. The instantaneous maximum of these 11

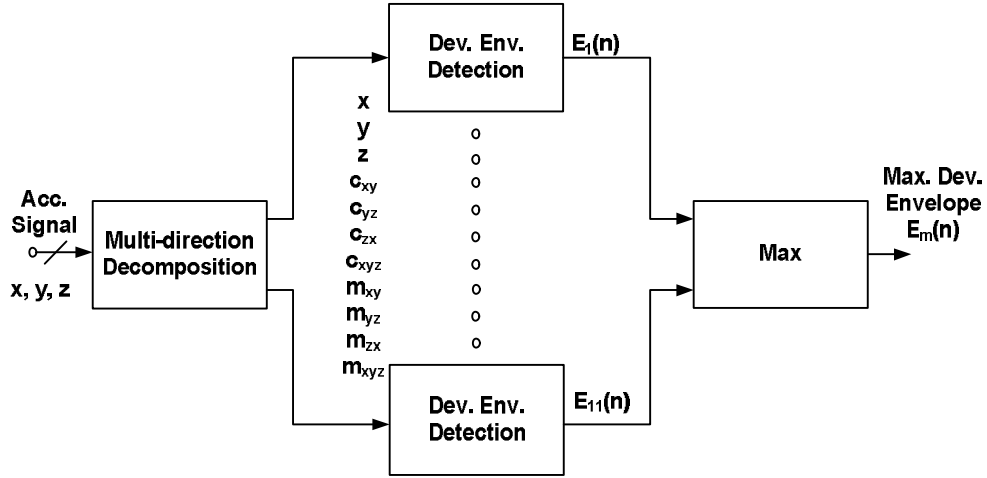


Figure 5.13: Signal processing for detection of maximum deviation envelope.

deviation envelopes is taken as the orientation-independent deviation envelope $E_m(n)$ as the processed output signal applied as the input to the threshold-based fall detection, as shown in Figure 5.13, with each 'deviation envelope detection' block corresponding to the signal processing as shown in Figure 5.12.

The threshold-based fall detection algorithm as proposed earlier by Kumar [28] for each of the components, is applied on $E_m(n)$. If $E_m(n)$ remains supra-threshold for duration greater than t_1 but not greater than t_2 , then fall is predicted. The algorithm is shown in Figure 5.14. The results from 80 simulated fall and 32 ADL (8 activities recorded for 4 volunteers) are analyzed to get the optimal values of the fall detection parameters as $\theta = 15$, $t_1 = 30$ ms, and $t_2 = 150$ ms.

The operations of signal acquisition from the sensor using burst-mode with polling, storage of the data in the flash memory, signal processing for finding $E(n)$ for the acceleration data, and the threshold-based fall detection algorithms were implemented for real-time processing on the microcontroller of the device. The sorting operations were carried out on integer variables, while float variables were used for other calculations. The total time required per sample is measured as 8 ms, which is within the available 9.31 ms for sampling frequency of 100 Hz.

5.5 Testing of the device with real-time processing for fall detection

The battery-powered device with the fall detection technique using the accelerometer data as described in the previous section implemented for real-time processing on its microcontroller was extensively tested under ADL and fall conditions. It was found to successfully detect the falls and generate the wireless alert message. Further testing with real-life use needs to be carried out.

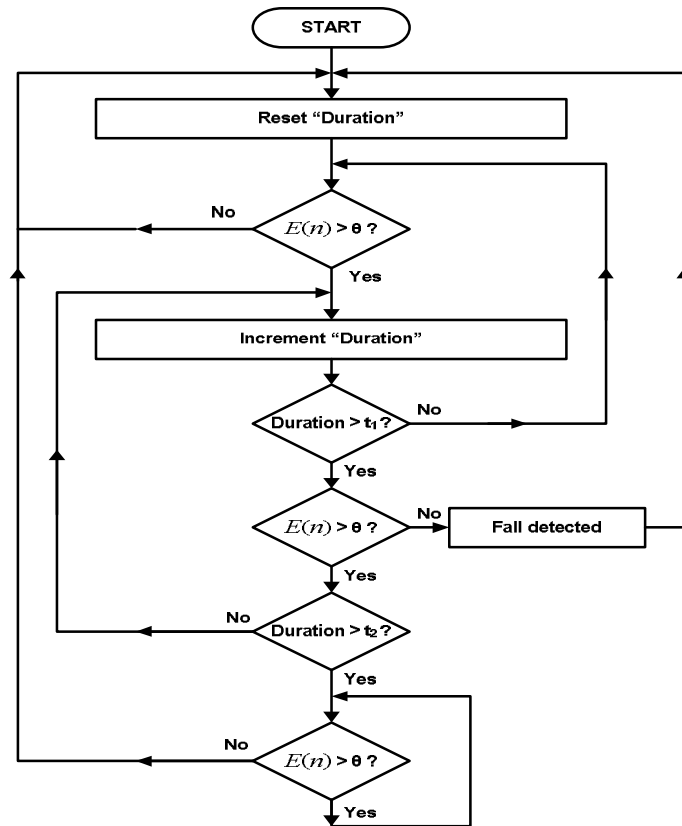


Figure 5.14: Flowchart of threshold-based real-time fall detection algorithm.

Chapter 6

SUMMARY AND CONCLUSION

Various aspects of the fall detection have been studied and compared. A battery-powered wearable fall detection module has been developed with an integrated sensor, microcontroller, Bluetooth transceiver, flash memory, and battery charger. The integrated sensor "MPU-9250" has tri-axial accelerometer, tri-axial gyroscope, and tri-axial magnetometer. The circuits inside the module work with 3 V supply and maximum current consumption of approximately 60 mA. The supply is obtained using a regulator and chargeable battery. With signal acquisition at 100 Hz, the module has internal memory for logging all the variables for over an hour. The hardware of the module has been tested and GUI has been developed to record the sensor data for various fall and non-fall activities. For data acquisition three modes were developed as (a) sample-by-sample mode, (b) interrupt based burst-mode, and (c) polling based burst-mode. The polling based burst-mode was found to be suitable for real-time fall detection. The signals from all three sensors are logged for offline processing. The processor of the module generates fall alerts over the Bluetooth link on the basis of real-time processing of the accelerometer data.

The device was used to record sensor outputs for the activities of daily life and simulated falls with different orientations of the device. The recorded results were analyzed to investigate the signal processing technique required to discriminate between fall and ADL. To make fall detection independent of device orientation, a set of 11 variables comprising the vector components along seven directions ($x, y, z, c_{xy}, c_{yz}, c_{zx}, c_{xyz}$) and four magnitudes ($m_{xyz}, m_{xy}, m_{yz}, m_{zx}$) is considered. The baseline of each variable is estimated using a median filter and deviation from the baseline is calculated by subtracting the baseline from the corresponding variable. Further the deviation envelope is detected by taking absolute value and low pass filtering using a median filter. It is observed that 15-point median filter is suitable for baseline estimation and 5-point median filter is suitable for envelope detection. In place of 15-point median which is used for the baseline estimation, the dynamic quantile tracking technique is used to significantly reduce the computation time. Further, the quick sort technique is used to implement the 5-point median filter which is used for envelope detection. It is found that squaring the input signal before processing for baseline and deviation improves the contrast between fall and ADL.

The deviation envelope detection is implemented for each of the eleven variables. The instantaneous maximum of these envelopes is considered as orientation-independent deviation $E_m(n)$ and is used for threshold-based fall detection. If $E_m(n)$ remains supra-threshold for a duration more than t_1 but not greater than t_2 , then fall is predicted. The results

from 80 simulated fall and 32 ADL are analyzed to get optimal values of the fall detection parameters as $\theta = 15$, $t_1 = 30$ ms, and $t_2 = 150$ ms.

For further improvement, the FDM hardware needs to be redesigned using a microcontroller with higher system clock or higher processing capacity. The PCB needs to be made smaller in size and the device consisting of board and the battery needs to be packaged with due consideration to aesthetics and usability. In the developed prototype the fall detection is carried out using the accelerometer and outputs of all the three sensors are logged for offline processing. Real-time processing of other two sensors for fall detection needs to be examined.

Appendix A

COMPONENT LIST

Table A.1: Component list of the FDM.

Component designator	Component description	Part number / Value	Quantity
C1, C2, C3, C4, C7, C8, C9, C11, C13, C15	Capacitor, ceramic, surface mounted	0.1 μ F	10
C14, C16, C6	Capacitor, ceramic, surface mounted	10.0 μ F	3
C10, C12	Capacitor, ceramic, surface mounted	1.0 μ F	2
C5	Capacitor, ceramic, surface mounted	10 nF	1
R1, R2	Resistor, surface mounted	4.7 k Ω	2
R4, R5	Resistor, surface mounted	220.0 Ω	2
R3	Resistor, surface mounted	100.0 Ω	1
R6, R7	Resistor, surface mounted	1k Ω	2
D1, D4	LED, surface mounted	GREEN	2
D2	LED, surface mounted	RED	1
D3	LED, surface mounted	YELLOW	1
CON1	Connector, 5-pin	CON1	1
CON2	Connector, 3-pin	CON2	1
SW	SPDT	SW	1
U1	IC, serial flash, 8 pin, SOIC	SST26VF064B	1
U2	Bluetooth module	RN42	1
U3	IC, 9-axis, accelerometer and gyroscope, magnetometer, 24 pin, QFN	MPU-9250	1
U4	IC, microcontroller, 44-pin, TQFP	24FJ64GB004	1
U5	IC, battery charger, 10 pin, DFN	MCP73833	1
U6	IC, Low dropout voltage regulator, 5 pin, SOT-23	MCP1802T-3002I/OT	1

Appendix B

SCHEMATIC DIAGRAM OF FDM

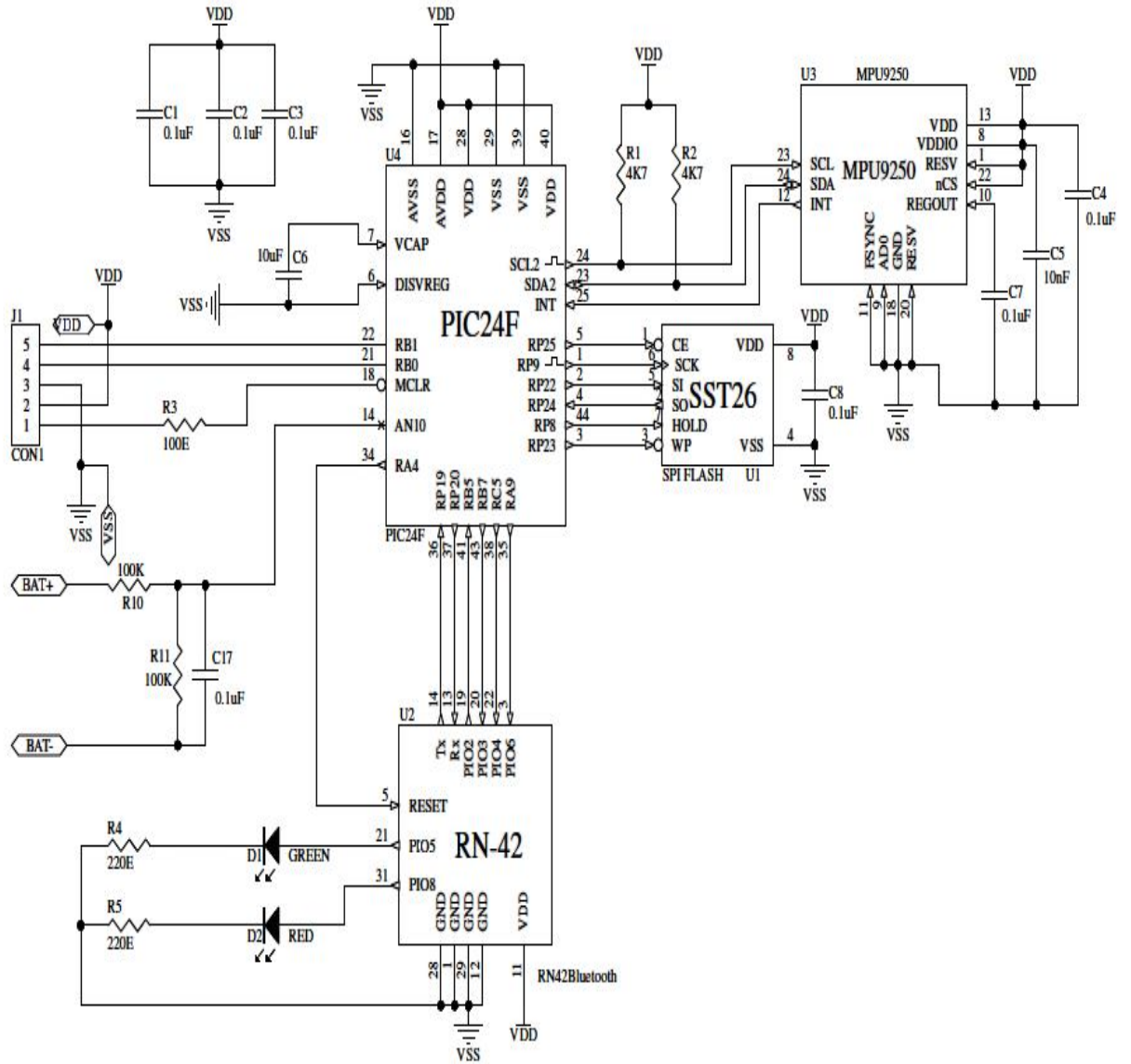


Figure B.1: Main schematic.



Appendix C

PCB LAYOUT

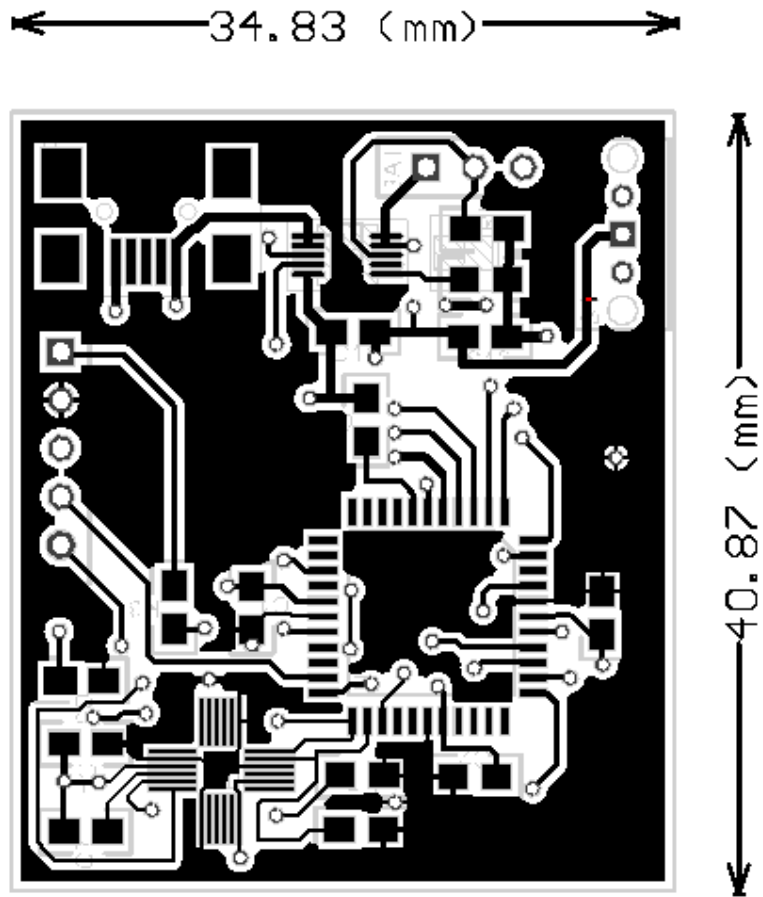


Figure C.1: Top layer of PCB.

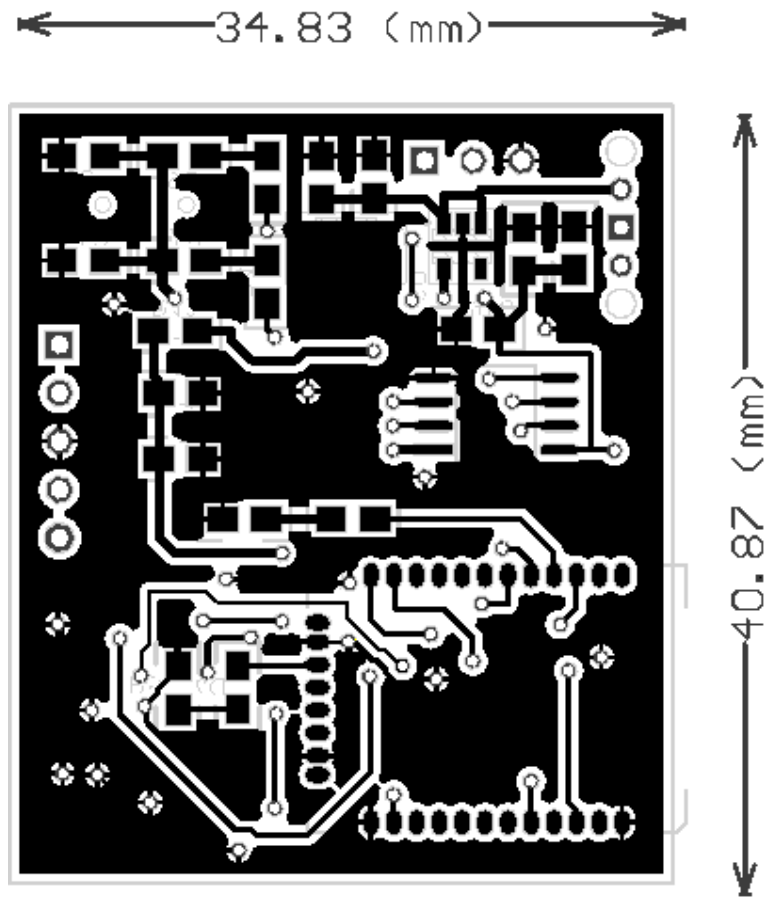
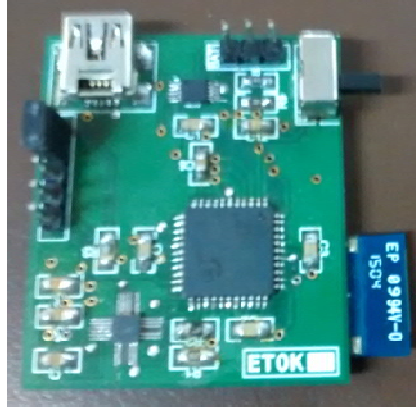


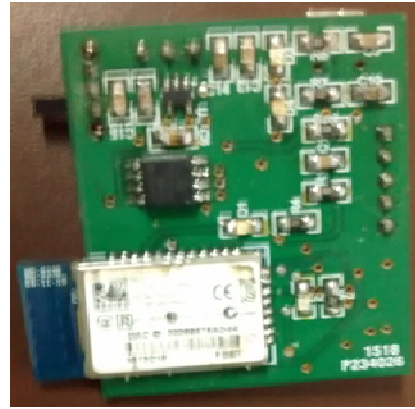
Figure C.2: Bottom layer of PCB.

Appendix D

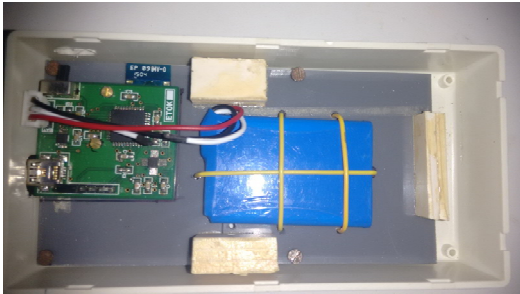
IMAGES OF FDM PROTOTYPE



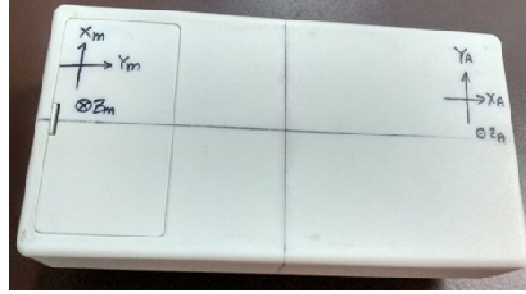
(a) Top view of PCB with components.



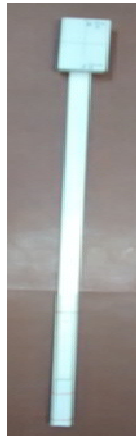
(b) Bottom view of PCB with components.



(c) Mounting of PCB and battery.



(d) Packaged FDM.



(e) FDM fixed at top of stick to record simulated falls.



(f) Wooden box fabricated for simulated fall with brackets to hold FDM in different orientations.

Figure D.1: Images of FDM prototype.

Appendix E

GIU DEVELOPED FOR DAQ

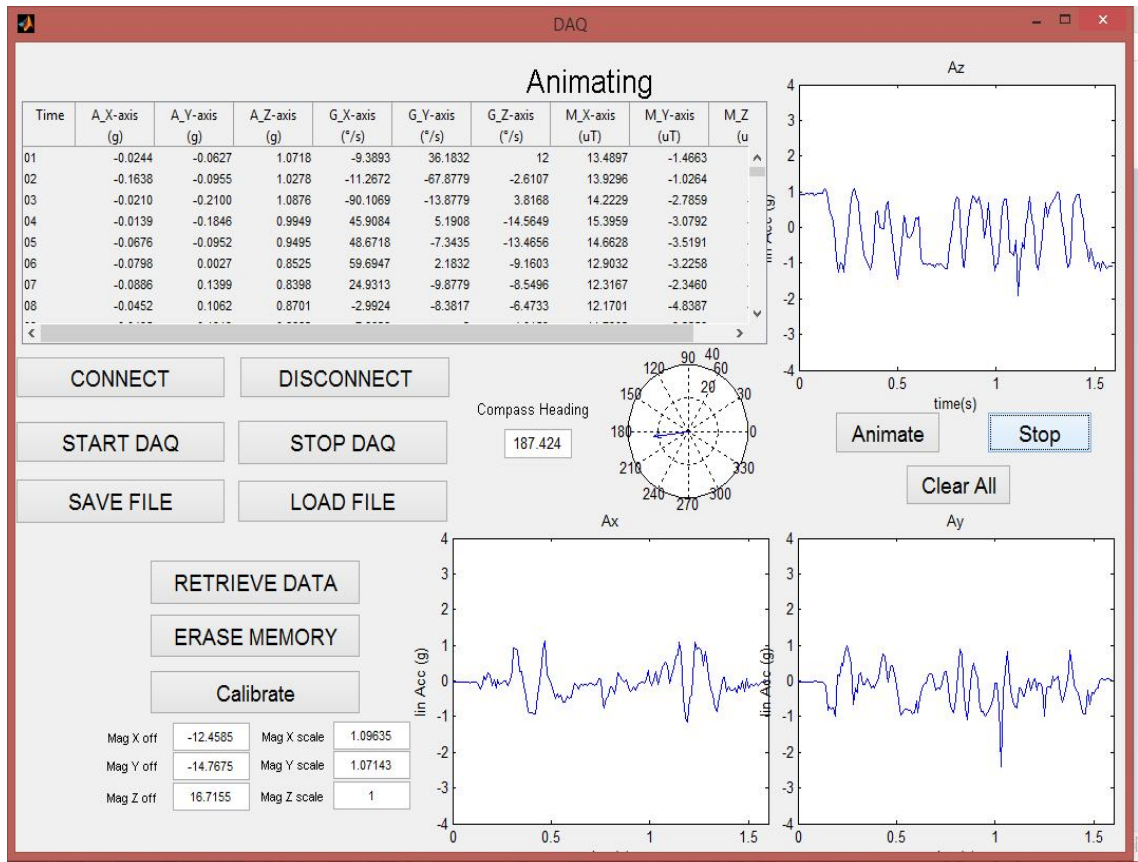


Figure E.1: Screenshot of GUI for DAQ (developed using MATLAB).

Appendix F

PLOTS OF ADL DATA

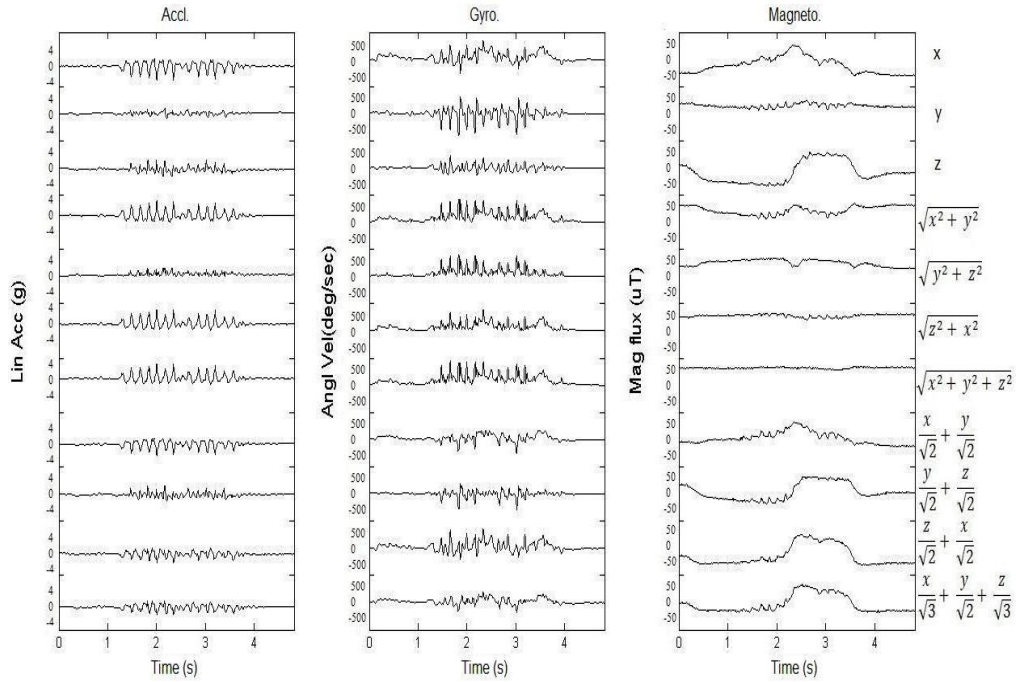


Figure F.1: Plots of ADL data: running.

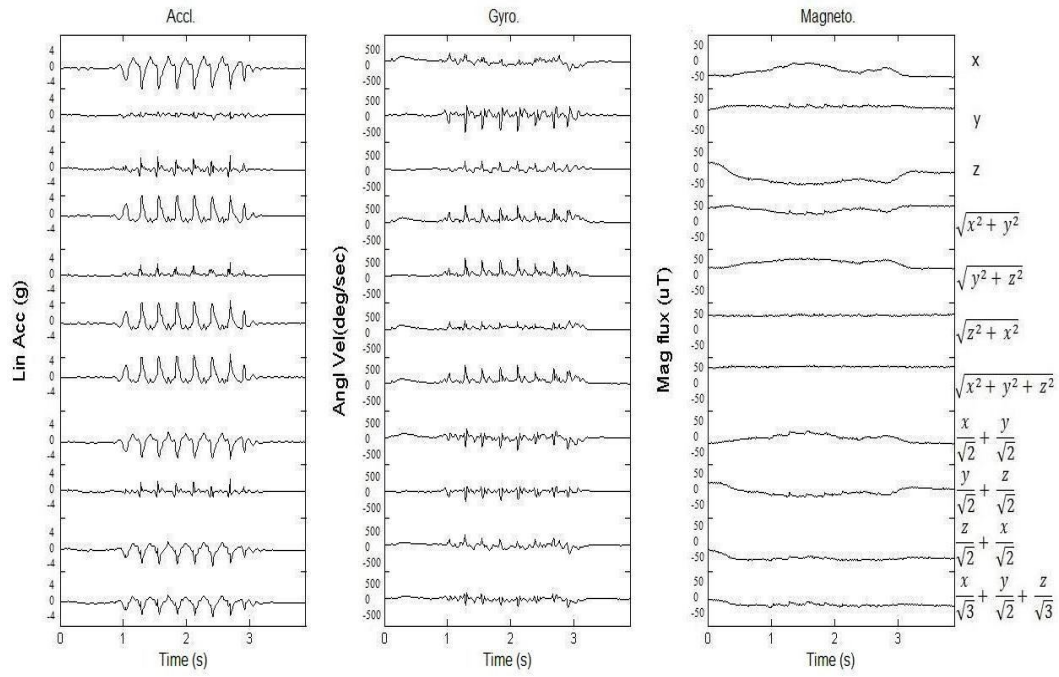


Figure F.2: Plots of ADL data: jumping.

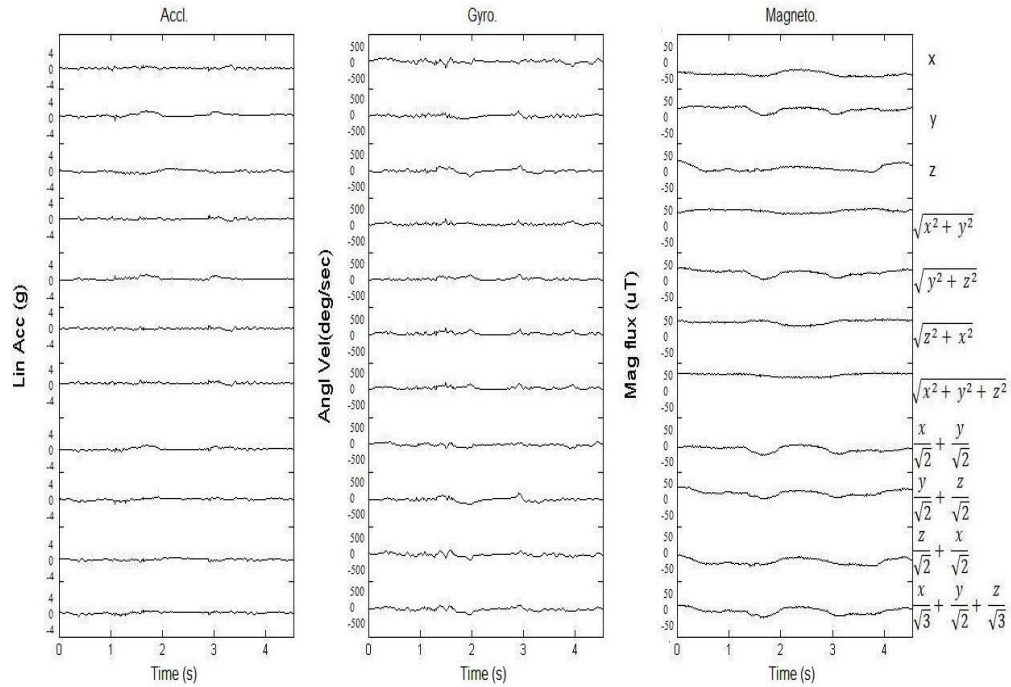


Figure F.3: Plots of ADL data: sitting down on a chair and getting up.

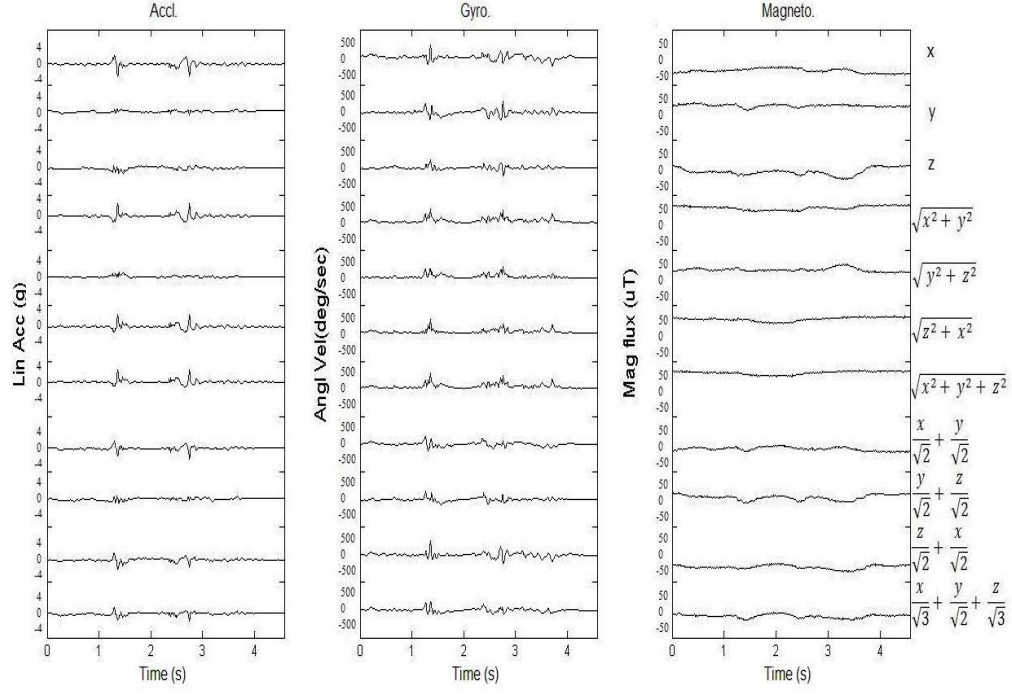


Figure F.4: Plots of ADL data: sitting down on a chair and getting up with jerk.

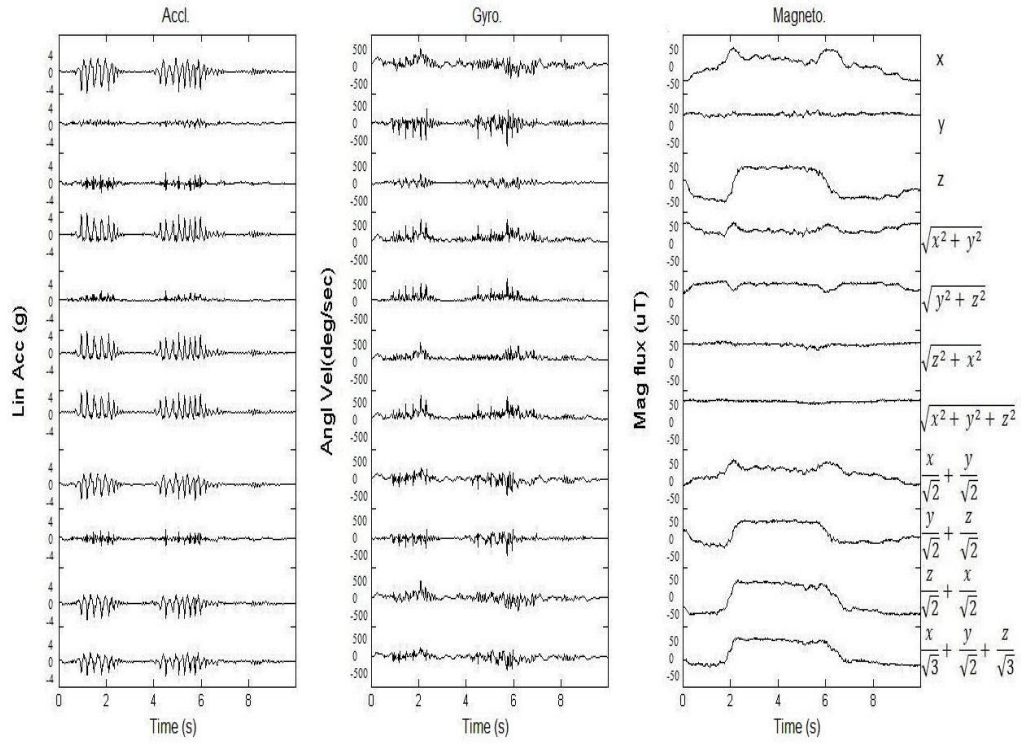


Figure F.5: Plots of ADL data: hopping.

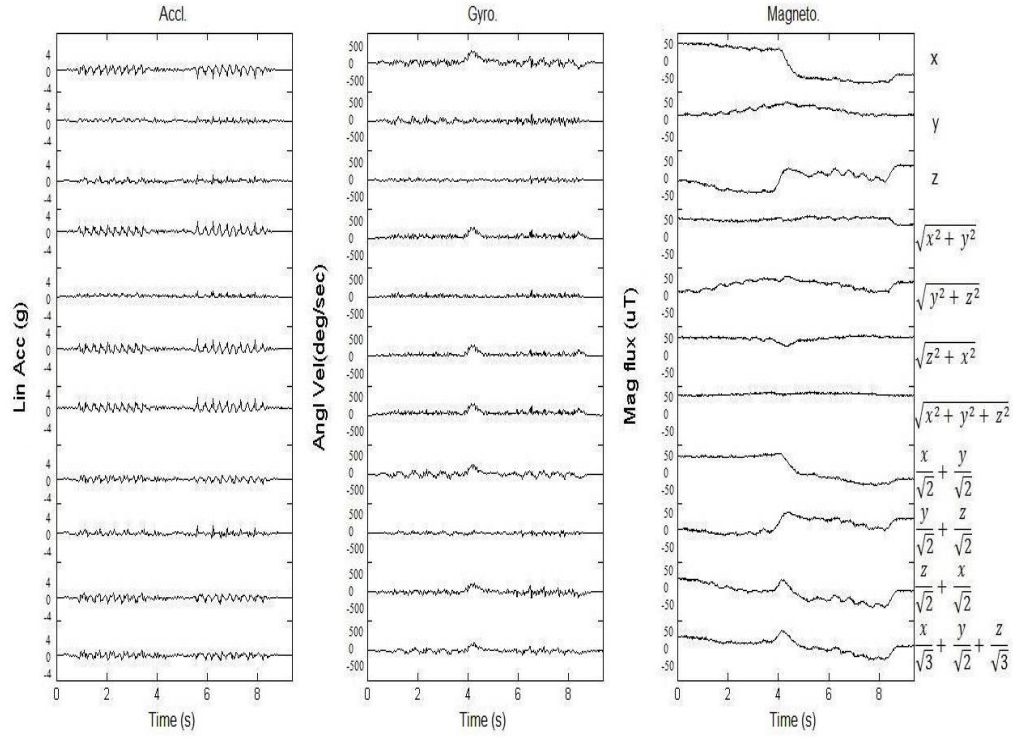


Figure F.6: Plots of ADL data: walking up stairs and down stairs.

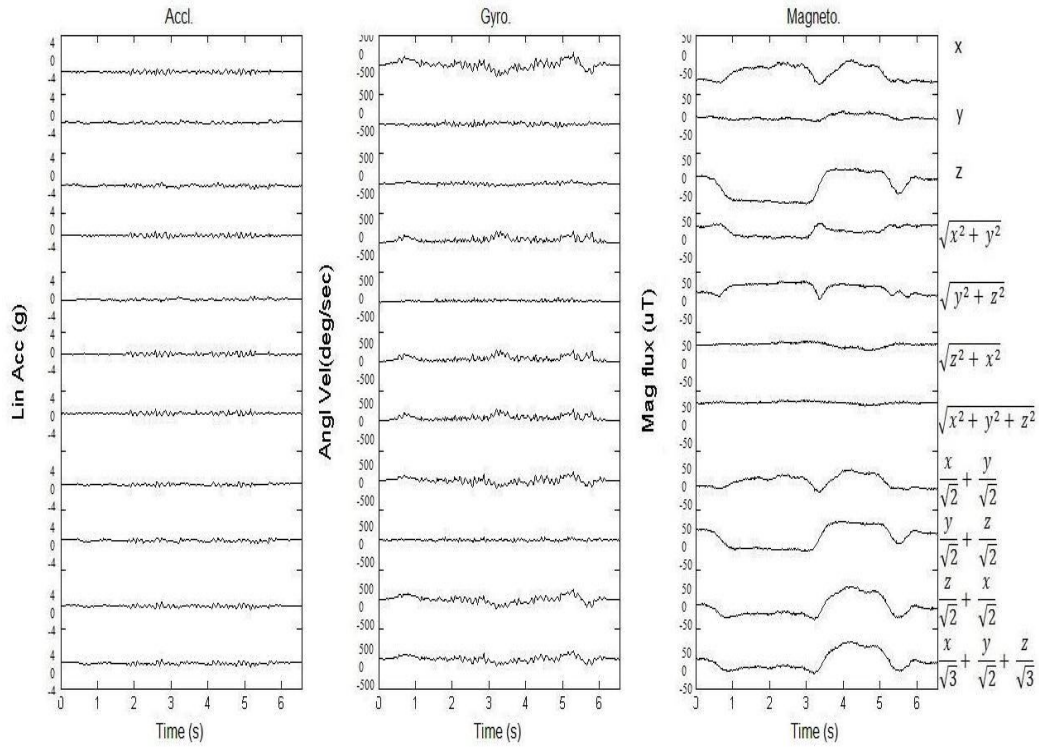


Figure F.7: Plots of ADL data: walking.

Appendix G

PLOTS OF FALL DATA

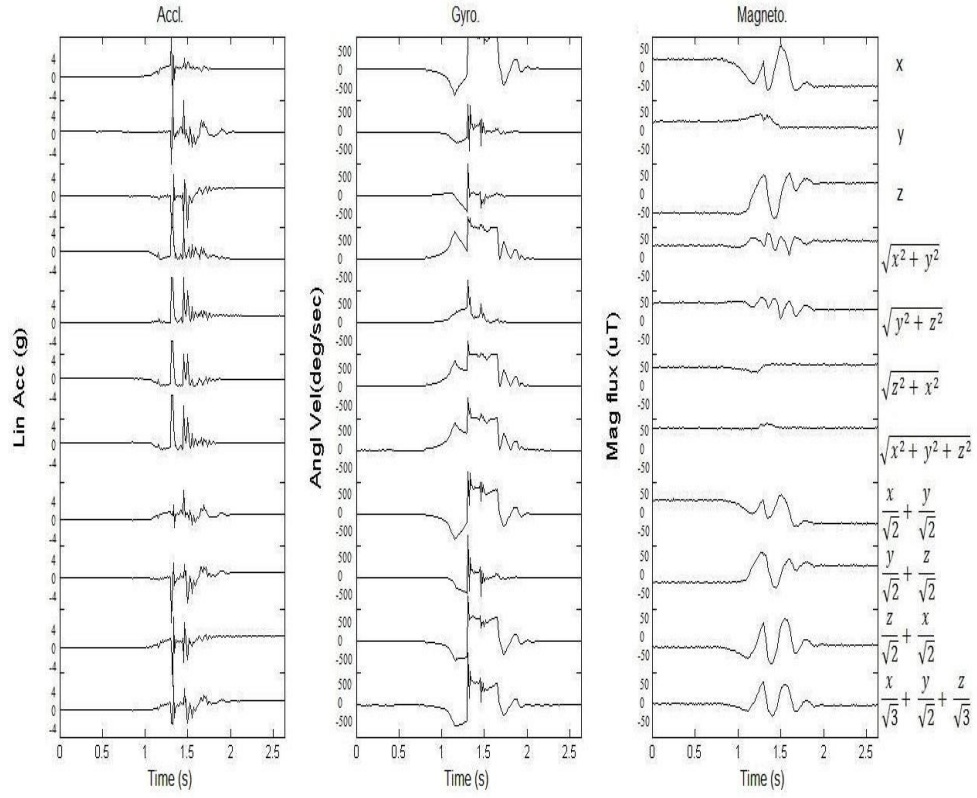


Figure G.1: Plots of data for simulated fall using the waist-height stick, with the device facing to the north and falling on the right with twist.

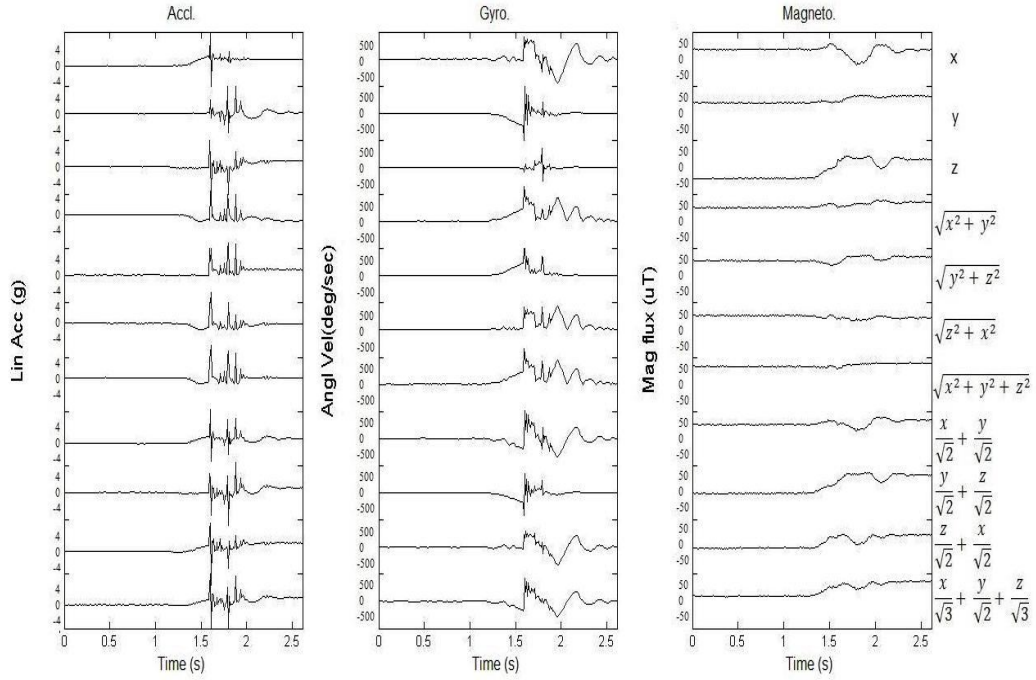


Figure G.2: Plots of data for simulated fall using the waist-height stick, with the device facing to the north and falling backward.

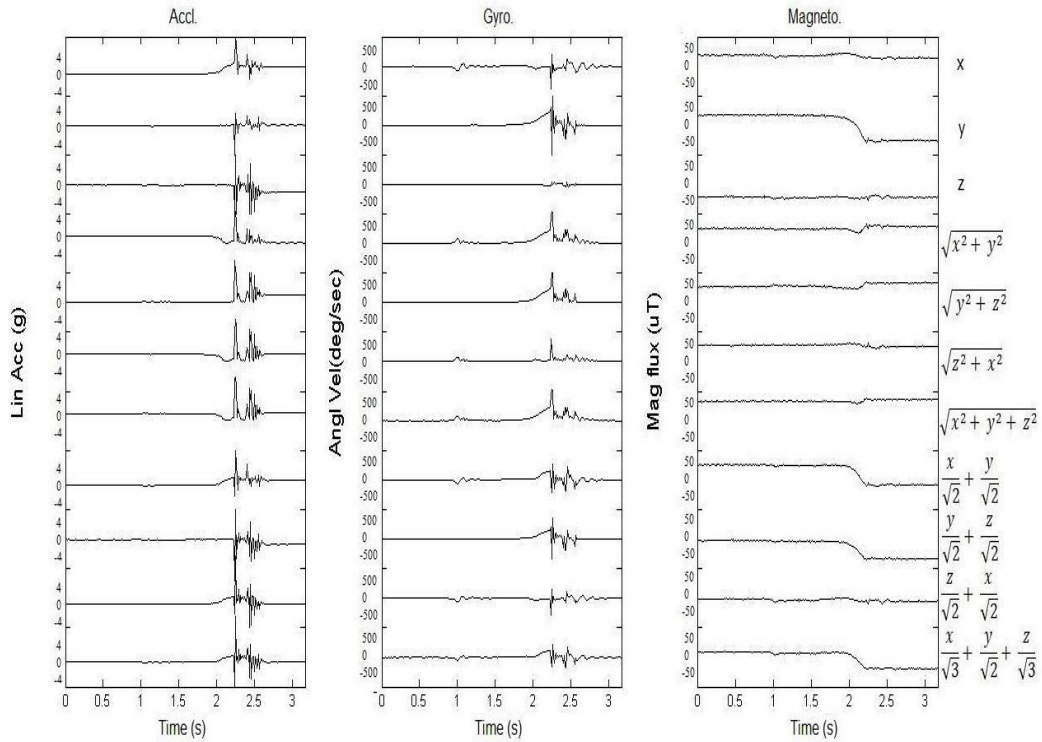


Figure G.3: Plots of data for simulated fall using the waist-height stick, with the device facing to the north and falling forward.

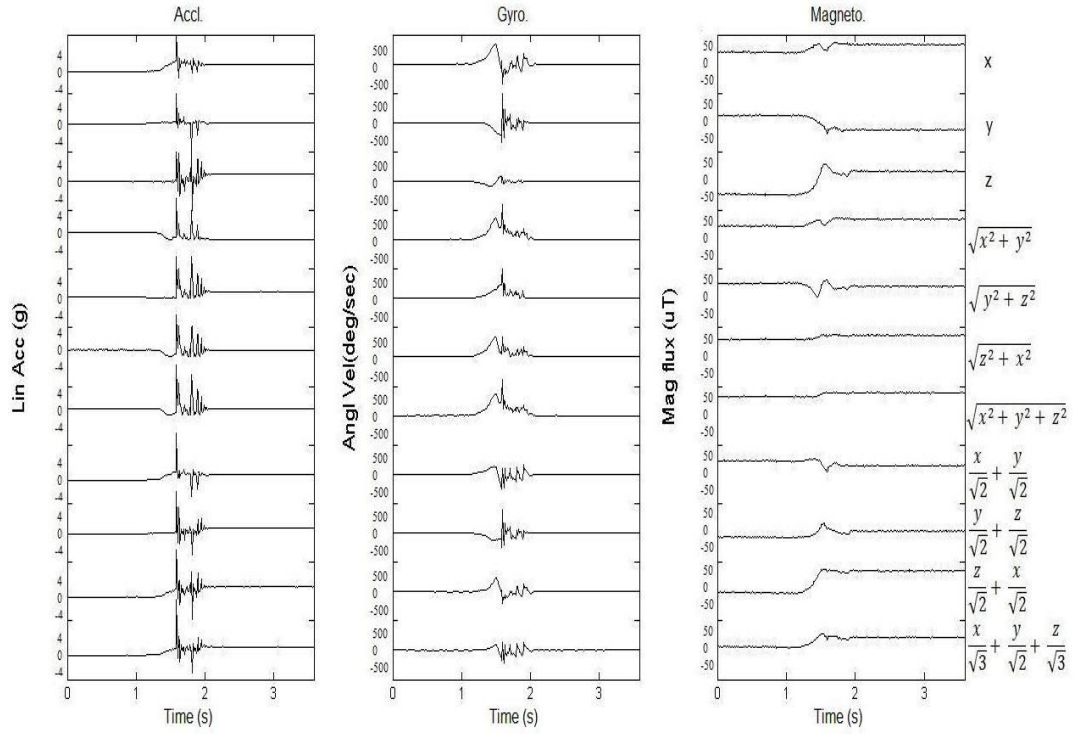


Figure G.4: Plots of data for simulated fall using the waist-height stick, with the device facing to the north and falling on the left with twist.

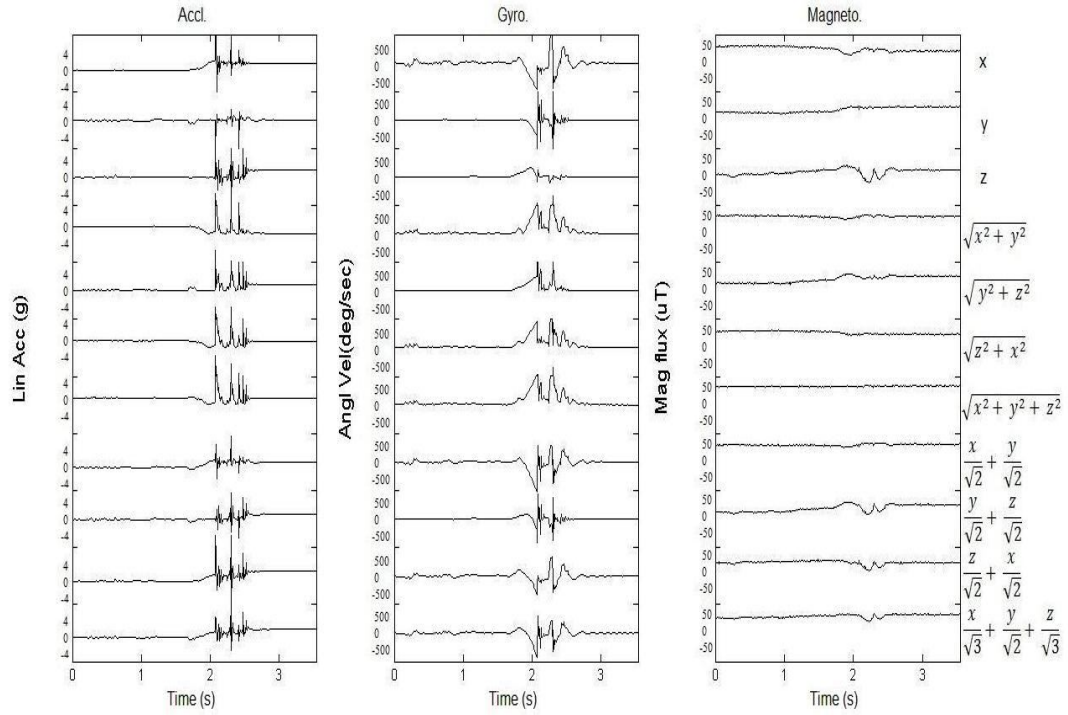


Figure G.5: Plots of data for simulated fall using the waist-height stick, with the device facing to the east and falling on the right with twist.

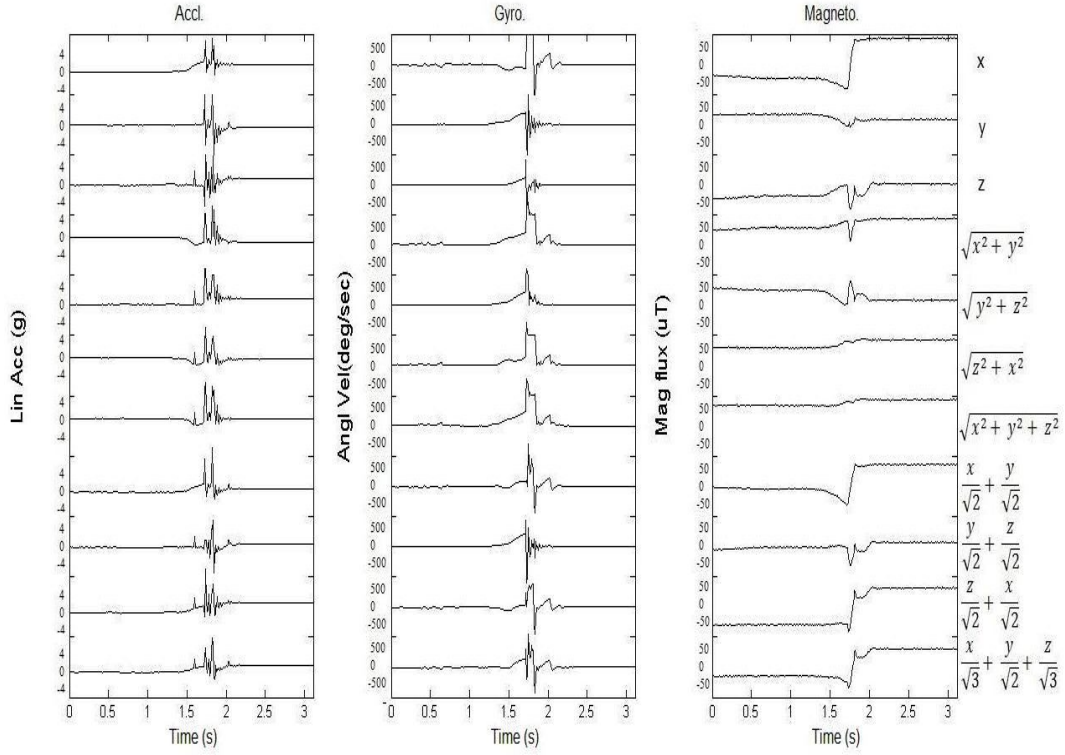


Figure G.6: Plots of data for simulated fall using the waist-height stick, with the device facing to the east and falling backward with twist.

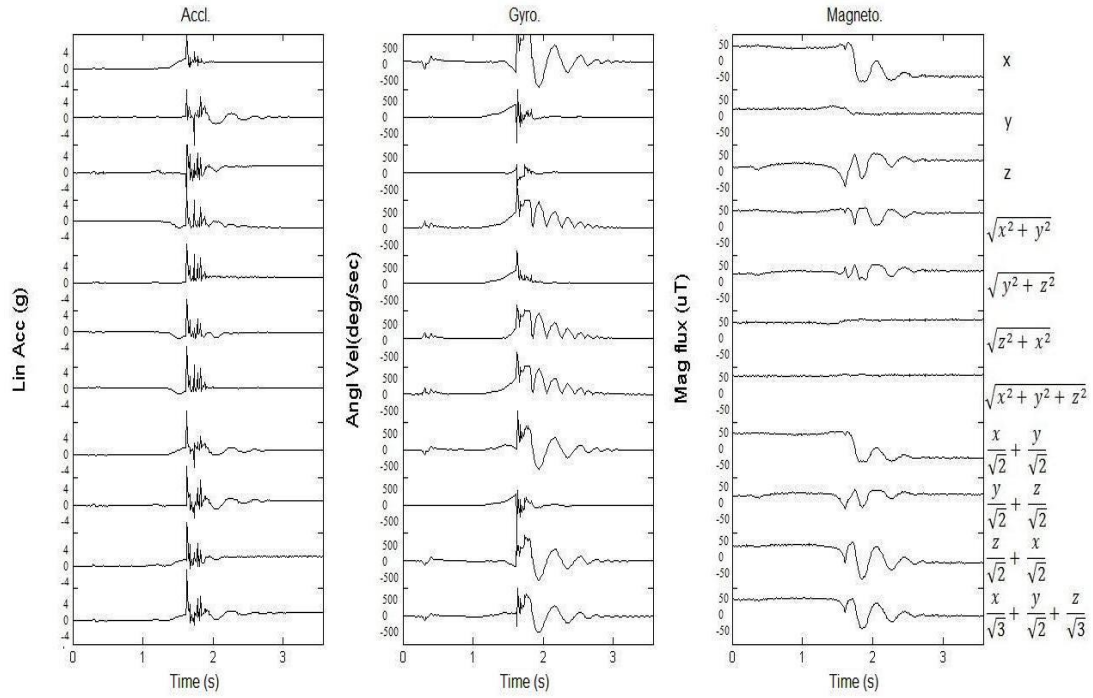


Figure G.7: Plots of data for simulated fall using the waist-height stick, with the device facing to the east and falling forward with twist.

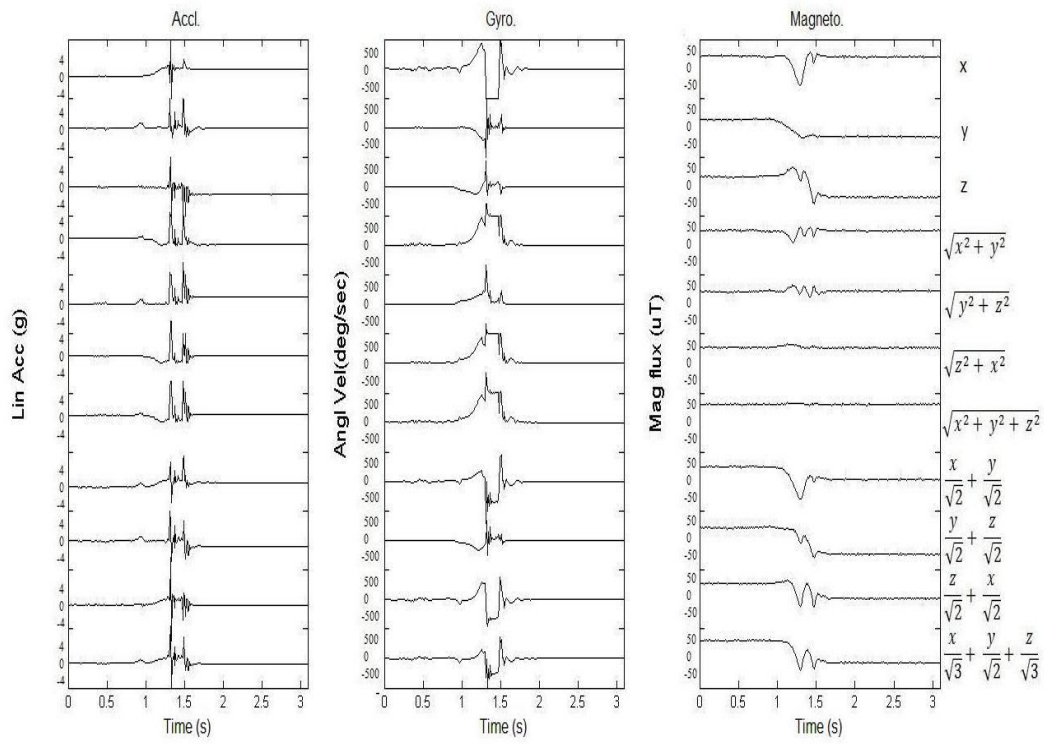





Figure G.8: Plots of data for simulated fall using the waist-height stick, with the device facing to the east and falling on the left with twist.

Appendix H

COMMERCIALLY AVAILABLE SYSTEMS FOR FALL DETECTION

Some of the fall detection systems which are commercially available are summarized in Table E.1. All these are wearable models which look like a pendant. They all have false alarm button which is used to confirm whether user is in need of assistance or not. If fall is detected and the person wearing the device does not respond to alarm via press of button in certain time, then it considers that user is not in condition of conformation and alerts the caretakers over cellular or telephone network.

Table H.1: Comparison of commercial products [15].

			
Product	Medical Guardian FallAlert	Philips Lifeline AutoAlert	MobileHelp Fall Button
Monthly charges	\$44.95 / Month	\$44.95/Month	\$44.95/Month
How can the device be worn	As a Pendant or on the hip	Pendant	Pendant
Telephone line requirement	No. Uses cellular network	Yes. Need telephone	No. Uses cellular network

References

- [1] Y. Wang and X. Y. Bai, "Research of fall detection and alarm applications for the elderly," in *Proc. Int. Conf. on Mechatronic Sciences, Electric Engineering and Computer (MEC) 2013*, Shenyang, China, pp. 615-619.
- [2] A. K. Bourke, J. V. O'brien, and G. M. Lyons, "Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm," *Gait & Posture*, vol. 26 (2), pp. 194-199, 2007 .
- [3] D. V. Nguyen, M. T. Le, A. D. Do, H. H. Duong, T. D. Thai, and D. H. Tran, "An efficient camera-based surveillance for fall detection of elderly people," in *Proc. 9th IEEE Conf. on Industrial Electronics and Applications*, Hangzhou, China, pp. 994-997, 2014.
- [4] J. Y. Hwang, J. M. Kang, Y. W. Jang, and H. C. Kim, "Development of novel algorithm and real-time monitoring ambulatory system using Bluetooth module for fall detection in the elderly," in *Proc. of the 26th Int. Conf. IEEE, Eng. Med. Biol. Soc. (EMBS 2004)*, vol. 1, San Francisco, California, pp. 2204-2207, 2004.
- [5] J. Cheng, "A framework for daily activity monitoring and fall detection based on surface electromyography and accelerometer signals," *IEEE J. Biomed. Health Informatics*, vol. 17 (1), pp. 38-45, 2013.
- [6] Z. Yan, V. Subbaraju, D. Chakraborty, A. Misra, and K. Aberer, "Energy-efficient continuous activity recognition on mobile phones: an activity-adaptive approach," in *Proc. 16th Int. Symposium on Wearable Computers*, Newcastle, UK, pp. 17-24, 2012.
- [7] P. Kumar and P. C. Pandey, "A wearable inertial sensing device for fall detection and motion tracking," in *Proc. 10th Ann. Conf. IEEE India Council (Indicon 2013)*, Mumbai, paper no. 1084, 2013.
- [8] M. Kangas, A. Konttila, I. Winblad, and T. Jämsä, "Determination of simple thresholds for accelerometry-based parameters for fall detection," in *Proc. 29th Ann. Int. Conf. IEEE, Eng. Med. Biol. Soc. (EMBS 2007)*, Lyon, France , pp. 1367-1370, 2007.
- [9] A. Sixsmith and N. Johnson, "A smart sensor to detect the falls of the elderly," *IEEE J. Pervasive Computing*, vol. 3 (2), pp. 42-47, 2004.
- [10] M. Tolkiehn, L. Atallah, B. Lo, and G. Z. Yang, "Direction sensitive fall detection using a triaxial accelerometer and a barometric pressure sensor," in *Ann. Int. Conf. IEEE, Eng. Med. Biology Soc. (EMBC 2011)*, Boston, Mass., pp. 369-372, 2011.
- [11] A. Gallagher, Y. Matsuoka, and W. T. Ang, "An efficient real-time human posture tracking algorithm using low-cost inertial and magnetic sensors," in *Proc. Int. Conf. IEEE/RSJ Intelligent Robots and Systems*, Sendai, Japan, vol. 3, pp. 2967-2972, 2004.

- [12] R. Zhu and Z. Zhou, "A real-time articulated human motion tracking using tri-axis inertial/magnetic sensors package," *IEEE Trans. Neural Systems Rehab. Eng.*, vol. 12 (2), pp. 295-302, 2004.
- [13] F. Bianchi, M. R. Narayanan, and B. G. Celler, "Falls event detection using triaxial accelerometry and barometric pressure measurement," in *31st Ann. Int. Conf. IEEE, Eng. Med. Biol. Soc.(EMBS 2009), Minneapolis, Minn.*, pp. 6111-6114, 2009.
- [14] Y. Li, G. Chen, Y. Shen, Y. Zhu, and Z. Cheng, "Accelerometer based fall detection sensor system for the elderly," in *Proc. Cloud Computing and Intelligent Systems (CCIS 2012)*, Hangzhou, China, pp. 1216-1220.
- [15] Medical Alert Advice, Available online: www.medicalalertadvice.com/fall-detection.php, Accessed: 25th October, 2015.
- [16] Q. Song, J. Feng, X. Zhang, and J. Xiao, "Obtaining the real time yaw angle using electronic compass in quadrotor systems," in *Proc. Int. Conf. System Science and Eng. 2012*, Dalian, China, pp. 238-242.
- [17] N. Tiwari and P. C. Pandey, "Speech enhancement using noise estimation based on dynamic quantile tracking for hearing impaired listeners," in *Proc. 21th Nat. Conf. Communications (NCC 2015)*, Mumbai, paper no. 1570056299, 2015.
- [18] InvenSense, "MPU-9250 Product specification revision 1.0," Available online: www.store.invensense.com/datasheets/invensense/MPU9250REV1.0.pdf, Accessed: 23th June, 2016.
- [19] Microchip, "PIC24fj64gb004 family datasheet 28/44 pin, 16 bit microcontrollers," Available online: www.microchip.com/downloads/en/DeviceDoc/39940d.pdf, Accessed: 2nd December, 2014.
- [20] Roving Networks, "RN42/RN42N class 2 Bluetooth module," Available online: www.microchip.com/downloads/en/DeviceDoc/rn-42-ds-v2.32r.pdf, Accessed: 12th March, 2015.
- [21] Microchip, "SST26VF064B/SST26VF064BA 3.0V serial quad I/O (SQI) flash memory," Available online: www.microchip.com/downloads/en/DeviceDoc/20005119D.pdf, Accessed: 12th March, 2015.
- [22] Microchip, "300 mA, high PARR, low quiescent current LDO," Available online: www.microchip.com/downloads/en/DeviceDoc/22053b.pdf, Accessed: 13th March, 2015.
- [23] Microchip, "MCP73833/4, Stand-alone linear Li-ion/Li-polymer charge management controller," Available online: www.microchip.com/downloads/en/DeviceDoc/22005a.pdf, Accessed: 13th March, 2015.

- [24] ST Microelectronics, “iNEMO inertial module: 3D accelerometer and 3D gyroscope,” Available online: www.st.com/content/ccc/resource/technical/document/datasheet/bd/61/af/53/b5/f5/4d/7b/DM00037200.pdf/files/DM00037200.pdf/jcr:content/translations/en.DM00037200.pdf, Accessed: 13th January, 2015.
- [25] NXP, “LPC-2148 single-chip, 16-bit microcontroller,” Available online: www.nxp.com/documents/data_sheet/LPC2141_42_44_46_48.pdf, Accessed: 23th June, 2016.
- [26] Texas Instruments, “MSP430f1611, 16-bit microcontroller,” Available online: www.ti.com/lit/ds/symlink/msp430f1611.pdf, Accessed: 23th June, 2016.
- [27] Texas Instruments, “CC2420, 2.4 GHz, IEEE 802.15.4, Zigbee-ready RF transceiver” Available online: www.ti.com/lit/ds/symlink/cc2420.pdf, Accessed: 23th June, 2016.
- [28] P. Kumar, “An inertial sensing module for movement and posture monitoring for assisted living,” *M.Tech. Dissertation*, Dept. Biosci. Bioeng., IIT Bombay, Mumbai, 2013.

Acknowledgements

I express my sincere gratitude to my respected guide Prof. P. C. Pandey, for his invaluable guidance during every stage of this project. I am thankful to Mr. Vidyadhar Kamble and Mr. Nandakumar Pai for their whole hearted support and encouragement. I am thankful to Mr. Chandrashekhhar Shele for his help during soldering the components on PCB and Mr. Sitaram Varak for making the wooden box used for testing of the device for simulated falls. I want to convey my gratitude towards my lab mates in Electronic Instrumentation Lab and Signal Processing and Instrumentation Lab for their help and support. I am thankful to Don Bosco Institute of Technology for financial support during the tenure of my M.Tech programme. Finally, I am thankful to my parents and family for their unconditional love and support.

Yogesh Gopinath Gholap.

June 2016

Author's Resume

Yogesh Gopinath Gholap: The author received B. E. degree in electronics engineering from Vivekanand Education Society's Institute of Technology, University of Mumbai, Maharashtra in 2009. Presently he is pursuing the M.Tech degree in electrical engineering in Indian Institute of Technology Bombay. His research interests include embedded system design and electronic instrumentation.