FALL DETECTION USING WAVELET TRANSFORM AND SUPPORT VECTOR MACHINE

Liang Hanghan

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Telecommunication Engineering

Suranaree University of Technology

Academic Year 2017

การตรวจจับการหกล้มด้วยการแปลงเวฟเล็ตและซัพพอร์ตเวคเตอร์แมชชีน

นางสาวเหลียง หังหัน

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมโทรคมนาคม
มหาวิทยาลัยเทคโนโลยีสุรนารี
ปีการศึกษา 2560

FALL DETECTION USING WAVELET TRANSFORM AND SUPPORT VECTOR MACHINE

Suranaree University of Technology has approved this thesis submitted in partial fulfillment of the requirements for a Master's Degree.

Thesis Examining Committee

(Assoc. Prof. Peerapong Uthansakul)

Chairperson

(Asst. Prof. Dr. Wipawee Hattagam)

Member (Thesis Advisor)

(Assoc. Prof. Dr. Chakchai So-In)

Member

(Prof. Dr. Santi Maensiri)

Vice Rector for Academic Affairs

and Internationalization

(Assoc. Prof. Flt. Lt. Dr. Kontorn Chamniprasart)

Dean of Institute of Engineering

เหลียง หังหัน : การตรวจจับการหกล้มด้วยการแปลงเวฟเล็ตและซับพอร์ตเวคเตอร์แมชชีน (FALL DETECTION USING WAVELET TRANSFORM AND SUPPORT VECTOR MACHINE) อาจารย์ที่ปรึกษา : ผู้ช่วยศาสตราจารย์ ดร.วิภาวี หัตถกรรม, 107หน้า.

การหกล้มนั้น ถือเป็นความเสี่ยงต่อชีวิตในผู้สูงอายุ การบาดเจ็บที่เกี่ยวข้องกับการหกล้มนั้น มีผลกระทบอย่างมากต่อชีวิตผู้สูงอายุ ยิ่งกว่านั้น หากผู้สูงอายุยังคงนอนอยู่ ภายหลังการหกล้ม โอกาสที่จะเกิดความภาวะแทรกซ้อนยิ่งเพิ่มสูงขึ้น จึงควรหลีกเลี่ยงภาวะแทรกซ้อนเหล่านั้นหากเป็นไปได้ ดังนั้น การศึกษาการตรวจจับการหกล้ม จึงมีความจำเป็นและควรแก่การศึกษาอย่างยิ่ง

ในการตรวจจับการหกล้มนั้น การใช้คุณลักษณะโดเมนเชิงความถี่ในการเคลื่อนไหวด้วยแรงเฉื่อยของร่างกาย ทำให้สามารถวิเคราะห์ความถี่ได้หลายระดับ อย่างไรก็ตาม การสกัดคุณลักษณะเชิงโดเมนความถี่นั้น มักใช้ความต้องการทางการคำนวณสูง วิทยานิพนธ์นี้ จึงเสนอวิธีการสกัดคุณลักษณะเชิงโดเมนความถี่ ที่ใช้การคำนวณต่ำ เรียกว่าการแปลงเวฟเล็ตแบบลิฟทิง (Lifting Wavelet Transform, LWT) ซึ่งให้การคำนวณอย่างมีประสิทธิภาพ เหมาะสมสำหรับอุปกรณ์กำลังงานต่ำ เช่น อุปกรณ์เซ็นเซอร์แบบสวมใส่เพื่อการตรวจจับการหกล้มในมนุษย์ คุณลักษณะซึ่งสกัดจาก LWT นั้น นำมาเป็นอินพุตของวิธีการแมชชีน เลิร์นนิง วิธีการหนึ่งเรียกว่าซับพอร์ตเวคเตอร์แมชชีน (Support Vector Machine, SVM) เพื่อระบุการหกล้มจากการเคลื่อนไหวในกิจกรรมประจำวันทั่วไป สมรรถนะของเวฟเล็ตแบบฮาร์ และ ไบออร์ทอกอนอล 2.2 ในระดับความถี่ต่างๆ ได้รับการเปรียบเทียบกับคุณลักษณะโดเมนเชิงเวลา ของรากเฉลี่ยกำลังสองของความเร่ง โดยใช้ชุดข้อมูลการหกล้มในมนุษย์ ผลการทดลองแสดงให้เห็นว่า คุณลักษณะสัมประสิทธิ์แบบละเอียดในระดับที่ 1 สำหรับเวฟเล็ตแบบฮาร์ และ ไบออร์ทอกอนอล 2.2 นั้น ได้ผลค่าความแม่นยำ ค่าความไว และค่าความจำเพาะในระดับดี

เพื่อการประเมินสมมรรถนะเพิ่มเติม จึงมีการเปรียบเทียบกับคุณลักษณะโดเมนเชิงความถี่ต่างๆ วิธีการ LWT ผนวกกับ SVM ที่นำเสนอ ถูกนำมาเปรียบเทียบกับการแปลงเวฟเล็ตแบบต่อเนื่อง (Continuous Wavelet Transform, CWT) ผนวกกับ SVM และประเมินผลในเทอมของความแม่นยำ ค่าความไว ค่าความจำเพาะ และและค่าการคำนวณเชิงเวลา จากผลการทดลองพบว่า LWT ด้วยเวฟเล็ตแบบฮาร์ ผนวกกับ SVM ให้ผลการทดลองเหนือกว่า CWT ผนวกกับ SVM โดยวิธีการนำเสนอนั้นให้ค่าการคำนวณเชิงเวลาต่ำกว่าวิธีการ CWT ผลองค์ความรู้และผลการค้นพบในวิทยานิพนธ์นี้ เป็นแนวทางในการประยุกต์ใช้คุณลักษณะโดเมนเชิงความถี่ที่มี

ประสิทธิภาพ ใช้ปริมาณการคำนวณต่ำ ในการตรวจจับการหกล้มแบบออฟไลน์ และสามารถนำมาใช้เป็นแนวทางเพื่ออ้างอิงในการตรวจจับการหกล้มแบบออนไลน์ต่อไปได้

สาขาวิชาวิศวกรรมโทรคมนาคม ลายมือชื่อนักศึกษา Luong Hanghan

ปีการศึกษา 2560 ลายมือชื่ออาจารย์ปรึกษา

LIANG HANGHAN : FALL DETECTION USING WAVELET TRANSFORM AND SUPPORT VECTOR MACHINE. THESIS ADVISOR : ASST. PROF. WIPAWEE HATTAGAM, Ph.D.,107 PP.

FALL DETECTION/SUPPORT VECTOR MACHINE (SVM)/WAVELET TRANSFORM/LIFTING WAVELET TRANSFORM (LWT)/ CONTINUOUS WAVELET TRANSFORM (CWT)/WEARABLE SENSORS

Fall has been life threatening for the elderly. The related injuries have a serious effect on their lives. Furthermore, if the elderly remains lying for prolonged time post fall, the chance of suffering from serious complications increases. Such complications should be avoided when possible. Thus, it is essential to study fall detection.

In fall detection, frequency domain feature of inertial body movement enables multi-resolution analysis. However, frequency domain feature extraction methods are typically computationally intensive. This thesis proposes a computationally light frequency domain feature extraction method based on lifting wavelet transform (LWT) which provides efficient computation suitable for low-powered devices such as wearable sensors for human fall detection. Features extracted LWT is then input into a machine learning method called support vector machine (SVM) to identify falls from activities of daily living. Performance of the Haar and Biorthogonal2.2 (Bior2.2) wavelets, under different multiresolution levels, are compared with the time domain feature of root-mean square acceleration using a dataset contains human falls. Results show that the 1-level-detail-coefficient features for both Haar and Biorthogonal 2.2 wavelets achieved good overall accuracy, sensitivity, and specificity.

In order to use SVM in a better way and further evaluate the performance of different frequency domain features, the use of SVM has been improved and the proposed LWT integrated with SVM algorithm has been compared with continuous wavelet transform (CWT) integrated with SVM. The performance has been evaluated in terms of accuracy, sensitivity, specificity and computational time. Results show that the proposed LWT with Haar wavelet integrated with SVM can outperform the CWT using customized wavelets and Haar wavelet, integrated with SVM. The proposed method also consumes less computational time than the CWT method. The contributions and findings in this thesis serve as guidelines for applying efficient and computationally light wavelet based features in off-line fall detection and maybe further used as a reference for on-line fall detection.

School of Telecommunication Engineering　　Student's Signature ______

Academic Year 2017　　Advisor's Signature ______

ACKNOWLEDGEMENT

First of all, I would like to show my deepest thanks to my thesis advisor, Asst. Prof. Dr. Wipawee Hattagam for her endless help and sincerely encouragement on every stage of writing this thesis. Her earnest and precise attitude toward work enlightens me most not only in this research but also in my future study. Without all her valuable instructions and impressive patient, I could not have completed my work this far.

I would also like to thank Assoc. Prof. Dr. Peerapong Uthansakul, Assoc. Prof. Dr. Chakchai So-In for all their kindness accepting to serve in my committee and for their apropos comments.

I am grateful to my teachers who have helped me to develop the fundamental and essential academic knowledge. My sincere appreciate goes to staff in the School of Telecommunication Engineering for their suggestions and support.

I shall extend my thanks to all the teachers and students from Suranaree University of Technology and Huazhong University of Science and Technology, who financially supported this work.

Last but not least, I would like to express my gratitude to all my friends for their support both in academic and in daily life.

Liang Hanghan

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

Page

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES (Continued)

Figure **Page**

LIST OF FIGURES (Continued)

Figure **Page**

SYMBOLS AND ABBREVIATIONS

SVM	=	support vector machine
LWT	=	lifting wavelet transform
CWT	=	continuous wavelet transform
TP	=	true positive
TN	=	true negative
RMS	=	root mean square
RSS	=	root sum square
SWT	=	stationary wavelet transform
STFT	=	short-time Fourier transform
UPV	=	upper peak value
LPV	=	lower peak value
ADLs	=	continuous of daily living
UFT	=	upper fall threshold
LFT	=	lower fall threshold
DT	=	decision tree
ANN	=	artificial neuron network
HMM	=	hidden Markov model
WT	=	wavelet transform
-1	=	negative, ADL

SYMBOLS AND ABBREVIATIONS (Continued)

+1	=	positive, fall
Tr	=	input training dataset
$[.]^T$	=	transpose
Th (.)	=	threshold function
$\boldsymbol{x_i}$	=	the i^{th} input feature
$y_i \in \{-1, +1\}$	=	label of feature
ω	=	normal vector of hyperplane
b	=	intercept
sgn (.)	=	sign function
x^+	=	positive support vector
x^-	=	negative support vector
d	=	distance between hyperplane and a support vector
s.t.	=	subject to
L (.)	=	Lagrange function
$\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_N]^T$	=	vector of Lagrange multipliers
∂	=	partial derivative
$\sum f(x)$	=	sum up function f
ξ_i	=	i^{th} scalar variables
C	=	regularization parameters

SYMBOLS AND ABBREVIATIONS (Continued)

$\phi(.)$	=	function that maps vector from input space to feature space
K(.)	=	kernel function
sv_{total}	=	root sum square of accelerometer data
SE	=	sensitivity
SP	=	specificity
AC	=	accuracy
x^i	=	input data point
x_{even}^i	=	i^{th} even sequence of a discrete signal
x_{odd}^i	=	i^{th} odd sequence of a discrete signal
cD	=	high frequency part (detail part) of a discrete signal
cA	=	low frequency part of a discrete signal
$\mathbf{A_i}$	=	the **i**th ADL
$\mathbf{F_i}$	=	the **i**th fall
Si	=	the **i**th data point of an average fall

CHAPTER I

INTRODUCTION

1.1 Background and related works

Nowadays, aging is an obvious tendency all over the world since the birth rate is decreasing. On one hand, global aging forces younger generation into working harder than ever before to support larger size of senior population. This tendency, on the other hand, leads to an increasing number of senior citizens living alone. By 2014, 9% of the senior citizens live alone and 19% of them live only with their spouse. Therefore, healthcare for elderly people is a serious social problem, especially in a country without a sound social security system. One of the biggest health threats of elderly who live alone are falls and related complications (Pierleoni, Pernini, et al., 2015). The Internet has brought forward potential applications which aid the elderly particular in fall detection. Information can be collected by wireless sensors such as accelerometer, gyroscope, magnetometer, and pressure sensors or the combination of this sensors. Considering the need of continuous measurement and ease of use, wearable sensor systems are one of the most promising systems.

1.1.1 Fall detection

When it comes to wearable sensor systems, several researchers have focused on processing the data collected from such sensors. They highlighted the need of pre-collected data to help with detecting new falls.

On the other hand, wearable sensors could be used to identify balance problems as well. This ability may lead to detecting falls and other disorders (Pierleoni, Pernini, et al., 2015) in advance. Thus, the purpose of wearable sensor systems can be divided into two categories: pre-fall detection and after-fall detection. Pre-fall detection is focused on balance monitoring and prevent any injury which may occur (Noshadi, Dabiri, Ahmadian, Amini, & Sarrafzadeh, 2013), (Paradiso, Hu, & Hsiao, 1999) when people falling down. Though the research objective was for dance movement not for fall detection, (Paradiso et al., 1999) was one of the earliest works about wearable sensor. It was a shoe-based posture recognition system which bridged the gap between human movements. (Noshadi et al., 2013) analyzed data in-depth and placed emphasis on images to detect abnormal balance before a fall happens.

As for after-fall detection, the aim is to detect falls without quick recovery and avoid long-lie on the ground which has a close relationship with after fall injury and mortality rate. Typically, those systems are evaluated by accuracy, sensitivity and specificity. Sensitivity is the ability of the algorithm to detect a fall (True Positive, TP), and specificity is the ability to distinguish the fall and non-fall (True Negative, TN)(Kianoush, Savazzi, Vicentini, Rampa, & Giussani, 2015). Though the two categories focus on different incidents, after-fall and pre-fall detection methods are similar in terms of data acquisition and feature extraction process.

1.1.2 Feature extraction in time domain

The two main assignments for fall detection, whether it is a pre-fall or post-fall detection, are feature extraction and decision making. Feature extraction is performed before decision making and has a significant influence on the decision.

Firstly, there are numerous amount of data, however, not all the data are equally important for fall detection. The part that is not as important as the other parts is considered a burden for any algorithm to calculate. Therefore, the challenge is to extract the features that contain appropriate information to represent fall movement.

Time domain features are the most traditional features for fall detection. The commonly extracted features are numerical features such as root mean square (RMS), root sum square, mean, variance. The data were collected by wearable sensors such as accelerometer, gyroscope, pressure sensor and fusion of such sensory data.

For instance, (Tang & Sazonov, 2014) presented a shoe-based posture recognition system. Common features such as mean, maximum, minimum, variance, standard and deviation of smart-shoe were extracted. Other features such as entropy and vertical direction were extracted depending on the actual situation in their work. Similar features were extracted in many related works, such as (Pierleoni, Pernini, et al., 2015) (Carlsson, 2015) (Özdemir & Barshan, 2014).

As for wearable sensors, data have been widely acquired in time-series. Time domain features are obtained from raw data directly. It is straightforward and easy to visualize.

However, time domain features have certain limitation in that it may not make full use of information by merely display observable trends (Banaee, Ahmed, & Loutfi, 2013). For example, according to time domain series, the amplitude change can be easily observed while the frequency change is not always clear. However, fall is a sudden change correspond to frequency change. Therefore, feature extraction in the frequency domain has to be investigated.

1.1.3 Feature extraction in frequency domain

Fall detection is regarded as a subset of human activities recognition. In this thesis, human activities are classified into two categories: activities of daily living (ADLs) and falls. Frequency domain feature extraction methods have been successfully applied to distinguish falls and ADLs in many works (Hanai, Nishimura, & Kuroda, 2009), (Yazar, Keskin, Töreyin, & Çetin, 2013), (Wójtowicz, Dobrowolski, & Tomczykiewicz, 2015). It displays the spectral domain information which may not be visually observed in time domain for fall detection. Typically, falls are often related to high frequency and ADLs are often related to periodic signals which has relatively lower frequency. Such frequency changes can be visualized after frequency analysis of the signal.

Frequency domain features were commonly extracted by Fourier transform based methods and wavelet transform based methods. (Lara & Labrador, 2013) summarized the main feature extraction methods using acceleration signal. Raw accelerometer signals are difficult to recognize since it contains high fluctuation. Thus, frequency transform methods such as Fourier transform (Björklund, Petersson, & Hendeby, 2015), discrete cosine transform and principle component analysis coefficients have been used to extract features from raw signals (He & Jin, 2009).

However, for fall detection, such algorithms have to encounter activities in different frequency bands. On the one hand, Fourier transform based methods are unsuitable for deal with non-stationary signals since it has a fixed window size. On the other hand, the computational complexity of Fourier transform (FFT) is $O(N\log_2 N)$ while that of discrete wavelet transform is $O(N)$ for N data points (Yazar et al., 2013). Wavelet transform based methods include continuous wavelet transform (CWT) and discrete wavelet transform (DWT). (Palmerini et al., 2015) described a CWT-based approach for fall detection. This work created a prototype wavelet of a typical fall pattern by the vector of average acceleration. By CWT, the degree of similarity of the new activity signals to the prototype was measured in terms of CWT coefficients. Another work in 2015 (Wójtowicz et al., 2015), compared performance of individual sensor and classifiers, accelerometer data with 5-level DWT and achieved 100% sensitivity, specificity and accuracy. Our work was inspired by these promising results. This thesis proposes the use of the lifting scheme of DWT which has been proposed to extract frequency domain features. Such features are input for the decision making algorithms for fall identification which are presented in the following section.

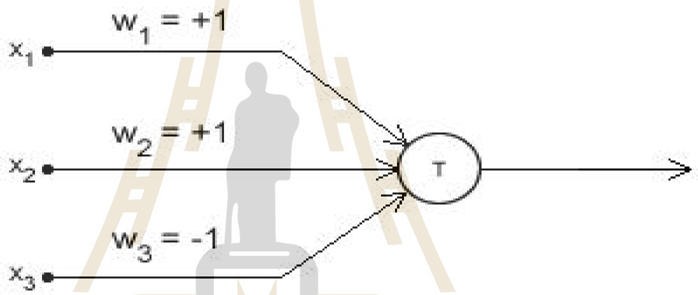
1.1.4 Decision making

Feature extraction enhances the domain features from wearable sensors. However, detecting a fall relies on decision making mechanism as well. In this section, decision making methods which regard extracted features mentioned above as input data are discussed. According to the same input data, different decision making methods or different models may make different decisions. There exist models that work better than the others for fall detection. Therefore, it worthwhile to have a close look at the decision making mechanisms for fall detection.

The threshold is one of the most basic methods for fall detection. (Bourke, O'brien, & Lyons, 2007) used the upper fall threshold (UFT) and lower fall threshold (LFT) to determine if a fixed threshold can distinguish falls and ADLs. The data used in (Bourke et al., 2007) were collected by wearable tri-axial accelerometer sensors that were attached to the trunk of young volunteers, respectively. Based on their results, UFT showed better results than LFT since it detected all ADLs. A comparison was made between this fixed-threshold method with machine learning methods in (Aziz, Musngi, Park, Mori, & Robinovitch, 2017). It was found that machine learning methods, especially support vector machine (SVM), offered an increased accuracy when the experiment was conducted in a laboratory with data from waist-mounted tri-axial accelerometers.

The decision tree method is a widely used machine learning methods. It is a tree structure model used to develop a classification rule that decides the class of any objectives. The decision tree consists of two parts: nodes and leaves. Nodes represent the attribute-test with a branch for each possible outcome. Leaves of the tree are class names, which are usually set as negative (-1) or positive (+1). The root node contains all the samples and those samples are divided into child nodes. From root nodes to leaves, there a series of decisions are made. The root should be the most robust attribute of the tree. If an error occurs earlier in a decision tree, more child nodes would classify samples based on the wrong decision, giving rise to a phenomenon called “error accumulation”.

The information entropy of a decision tree is a method to measure the degree of object class similar to each other (Myles, Feudale, Liu, Woody, & Brown, 2004). The more the number of objects belonging to same class, the smaller information entropy. Because the number of objects in each node is different, the ratio of number of objects in one node over the total number of nodes was used as a weight parameter for this node, to calculate information gain. Higher information gain means more objects which were classified by this attribute-based test belong to the same class. The ID3 decision tree is a well-known decision tree method that uses information gain as attribute-test (J. R.



$w_1 = +1$
x_1
$w_2 = +1$
x_2
T
$w_3 = -1$
x_3

layers called hidden layers between the input layer and output layer. Neural networks with hidden layers are called multi-layer feedforward neural networks. Neurons in both input/output layer and hidden layers are connected with neurons in neighboring layers. But the neurons in the same layer have no connection nor any cross-layer connection.

As indicated in (Z. Wang, Jiang, Hu, & Li, 2012), an incremental learning method based on neural network was proposed for ADL classification. Compared with other machine learning methods, for example, the decision tree, the training time of ANN was shorter while achieving high classification accuracy. However, (Parkka et al., 2006) stated that ANN may not be as stable as decision tree. This is because human activity monitoring data may be noisy for ANN. Furthermore, ANN may easily overfit without any protecting strategy for its learning ability.

Hidden Markov Model (HMM) is a type of probabilistic graphical model (Koller & Friedman, 2009). It is based on a dynamic Bayesian network. One of the earliest applications of HMM is for speech recognition (Baker, 1975). If a Markov process has N discrete states, the system transits among the N states according to certain transition probabilities. Different from a normal Markov model, the observation of a HMM state is a probabilistic function of this state. That is, the state of HMM cannot be observed directly (hidden). The hidden variables can only be observed from other stochastic processes (Rabiner, 1989). Sequences of human activities can be modeled as a Markov chain. One posture represents a state and movements from one posture to another posture are simulated as state transitions. (Tong, Song, Ge, & Liu, 2013) came up with a HMM-based fall detection and prediction algorithm using data collected by wearable sensors. The results showed that the HMM-based method can predict a fall event 200-400 ms ahead of the incident. Although not deployed for fall detection off-line, (Kianoush et al., 2015) applied HMM to trade-off between decreasing sampling rate and achieving the requirement of real-time fall detection system. Apart from traditional HMM, (Li et al., 2015) proposed an extended HMM to overcome the problem which HMM cannot handle large volume of data. Due to the high computational cost of HMM, the extended HMM in (Li et al., 2015) was shown to be a promising method for fall detection. The advantage lays on the ability to find out hidden or unexpected information from observed data for fall detection while the disadvantage is the high computational burden of this algorithm.

Support vector machine (SVM) is a popular machine learning method (Banaee et al., 2013). The idea of SVM is to map data points from input space to a feature space. A plane in feature space called a hyperplane or decision boundary is used to classify the samples into two regions (Pierleoni, Pernini, et al., 2015). The further away the samples are from the hyperplane, the less the classification error occurs. Therefore, a hyperplane needs to be placed in such a position that the distance between the boundary and the nearest sample (support vector) is maximum.

This method has been mathematically proven (Cortes & Vapnik, 1995) and was implemented in a convenient toolbox named LIBSVM (Chang & Lin, 2011). SVM was used successfully used in speech recognition (Ma, Randolph, & Drish, 2001), facial recognition (Heisele, Ho, & Poggio, 2001), stress and influenza classification (Wijaya, Prihatmanto, & Wijaya, 2016). SVM has also been used for fall detection based on wearable sensors (Pierleoni, Belli, et al., 2015), (Özdemir & Barshan, 2014) and (Liu & Cheng, 2012). SVM in (Pierleoni, Belli, et al., 2015) was used to find a proper hyperplane to detect falls based on acceleration by the training process. (Liu & Cheng, 2012) and (Özdemir & Barshan, 2014) investigated the computational cost of SVM by measuring training time and testing time. They came to a conclusion that SVM algorithm for fall detection has an acceptable performance. (Özdemir & Barshan, 2014) proposed an approach to reduce computational complexity of SVM in fall detection based on three tri-axial sensors (accelerometer, gyroscope, and magnetometer or compass). Because not all the collected data were equally important to detect a fall, (Özdemir & Barshan, 2014) extracted a part of features to reduce the volume of data input and decrease computational complexity. This was carried out by only focusing on features inside a 4-second window instead of inputting the entire activity into the algorithm directly.

SVM is a robust method for fall detection when compared with other methods such as threshold-based methods (Aziz et al., 2017) and decision tree methods (Özdemir, 2016) under the same circumstances. It was proved that performance of SVM cannot be improved straightforwardly by adding more sensors (Özdemir, 2016), changing the training or testing data size (Ustuner, Sanli, & Abdikan, 2016) or varying sensor locations (Shibuya et al., 2015). More specifically, (Shibuya et al., 2015) compared the effect of wearable sensor location on the performance of SVM. They found that no matter where the sensors were placed (i.e. on the back, chest or other body parts), SVM cannot effectively detect the "sliding" type of fall (for example, fall while sitting on a chair). As for the effect of training size on SVM, (Shibuya et al., 2015) found that SVM was robust against imbalanced training and testing data size in image classification. Later, (Özdemir, 2016) investigated the overall accuracy of SVM for fall detection using wearable sensors. They showed that SVM accuracy for fall detection did not significantly depend on the number or location of sensors. In particular, SVM achieved 99.27% accuracy with a single sensor located on the thigh whereas achieved

99.48% accuracy with data from 6 sensors placed on different locations of human body.

In terms of SVM algorithm itself, (Nukala et al., 2014) compared with linear, polynomial and radial basis function (RBF) kernel functions. RBF performs better wherever sensors were located and whatever was training and testing size. With so several factors affecting its performance, (Nukala et al., 2014) stated that though the original SVM is a widely used tool for fall detection and human posture recognition, it may not attain consistent detection accuracy. This is due to the existence of certain data points located near the hyperplane. Therefore, (Tang & Sazonov, 2014) proposed a SVM with data rejection for human postures and activities recognition. They measured the distance between samples and the hyperplane, samples that may be too close to the hyperplane were rejected. Furthermore, the result of (Tang & Sazonov, 2014) showed that by using the data rejection method, the mean accuracy rate increased by 17.5% with feature extraction and also increased about 2% without feature extraction. This is a considerable increase for fall detection algorithms. One possible explanation for this may be that some important information for accurate classification was covered by data positioned close to the hyperplane.

From previous works, it should be noted that most works which deployed SVM for fall detection used time-series features (Aziz et al., 2017), (Shibuya et al., 2015) and (Colkesen, 2012). To the best of our knowledge, only (Özdemir & Barshan, 2014) investigated the usage of spectral domain and time domain features together with SVM. The spectral features were the first 11 values of autocorrelation sequence and the first 5 peaks of the corresponding frequency after discrete wavelet transform (DWT). Their DWT performance showed a good accuracy, achieving more than 97% when the testing the data size was significantly larger than the training data size. Since the unbalanced training data and testing date size is usually the case in the real-world, it may be worthwhile to investigate other spectral features to be used with SVM. Therefore, the underlying objectives of this research are as follows.

1.2 Research objectives

1.2.1 To design a lightweight wavelet transform frequency domain feature extraction method integrated with an appropriate SVM model to detect falls based on a tri-accelerometer collected data.

1.2.2 To study the impact of wavelet transform methods on the performance of SVM.

1.2.3 To construct a fall detection based on lifting wavelet transform with lifting wavelet transform and SVM which has low computational requirement and performs well in terms of accuracy, specificity and sensitivity.

1.3 Research hypothesis

1.3.1 The proposed wavelet transform based algorithm works better than the traditional time domain algorithm for fall detection.

1.3.2 Wavelet transform achieves better results than time domain features, and LWT is even better than CWT in the same situation.

1.3.3 The suitable window length depends on frequency because fall occurs in a very short period of time.

1.3.4 Wavelet transform with threshold works better than wavelet alone.

1.3.5 CWT is more complex than LWT in terms of computational complexity.

1.4 Basic agreements

1.4.1 MATLAB R2014b @win10 was used to imply this experiment.

1.4.2 Data used in this work was collected by a single tri-accelerometer. Two different datasets were used, one is from (Kwolek, B., & Kepski, M., 2014) and the other one from Imperial College London (Pannurat, N., Thiemjarus, S., & Nantajeewarawat, E., 2017).

1.4.3 The dataset with video includes 40 activities of daily living (ADLs) and 30 falls (Falls) collected by a tri-accelerometer attached on waist.

1.4.4 The data set from Imperial College of London includes 13 types of falls and 12 types of ADLs conducted by 12 objects with a sensor attached on the waist.

1.5 Scope and limitation

1.5.1 Two different datasets were studied in this thesis.

1.5.2 The proposed algorithm is based on a single accelerometer data.

1.5.3 All the activities used in experiments were performed by young subjects.

1.5.4 Simulation is conducted by MATLAB and Python based on LIBSVM.

1.6 Expected benefits

1.6.1 To obtain computational light wavelet transform based features to detect falls based on SVM with data collected by a wearable tri-accelerometer.

1.6.2 To compare performance of the wavelet transform based features and the time domain features with SVM for fall detection.

1.7 Synopsis of thesis

The remaining parts of this thesis are organized as follows.

Chapter II illustrates the methodology used in this work. This chapter presents discrete wavelet transform with lifting scheme (LWT) and continuous wavelet transform (CWT). Then followed by the theory of support vector machine (SVM).

Chapter III presents performance evaluation based on the proposed LWT and SVM integration method. The performance is compared to that of a simple time domain feature. LWT provides computational efficiency that is suitable for on-board data processing and SVM is used as a fall identifier. The Haar and Bior2.2 mother wavelets for LWT are compared since they performed best. The best LWT multiresolution level of coefficients are analyzed in this chapter. Since SVM is used to classify sample points, a threshold was determined to classify falls from ADLs.

Chapter IV presents a performance evaluation based on another data. In particular, this chapter focused on the comparison between CWT and LWT feature based on the Haar and customized wavelets. The findings of this chapter emphasize that the proposed lifting scheme is computationally light and can outperform the higher computational continuous wavelet transform feature.

Chapter V concludes the thesis, highlighting the findings and contributions in chapter III and chapter IV.

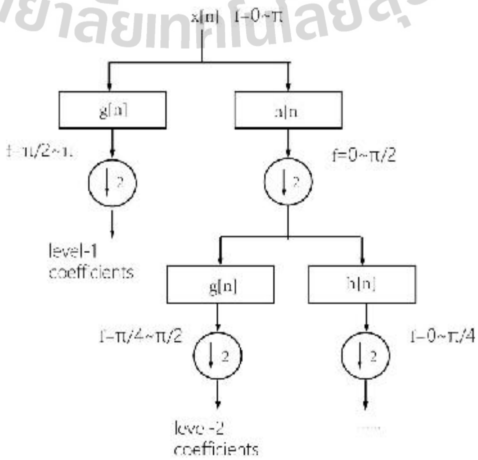
CHAPTER II

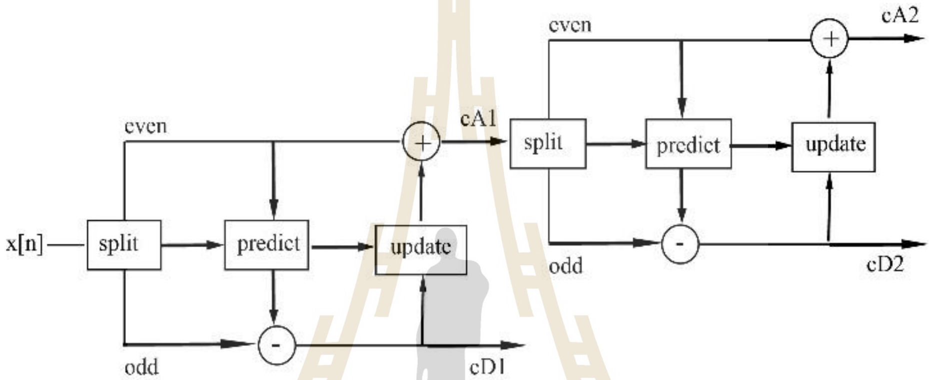
FUNDAMENTAL THEORY

SVM is a supervised machine learning method. The main steps can be simply concluded as training and testing. The input data is called features. The way we extract features from the original data has an impact on the performance of SVM. Since there are numbers of previous works discussed how to improve SVM itself, we focused on finding suitable specific features for fall detection. The two aspects used to analyze signals are usually time domain and frequency domain. Time domain features are features such as root-sum-square, maximum, minimum and so on. Frequency domain features may include CWT coefficients, LWT coefficients and maximum frequency. Therefore, in this chapter, SVM is first introduced. Followed by the introduction of CWT and LWT. At the end of this chapter, data preprocessing and classification details such as windowing, labeling and the way to calculate threshold was presented.

2.1 Lifting Wavelet Transform (LWT)

Discrete wavelet transform (DWT) comes later than CWT which will be introduced in next section. It went back to 1976 when A. Corosier and D. Esteban proposed a method to split channels (Esteban & Galad, 1976). One of the earliest applications of DWT was in speech recognition (Krishnan, Neophytou, & Prescott, 1994). Identically, DWT has advantages on resolution over STFT as CWT. DWT uses filters to split high or low frequency part of the signal and uses upsampling or

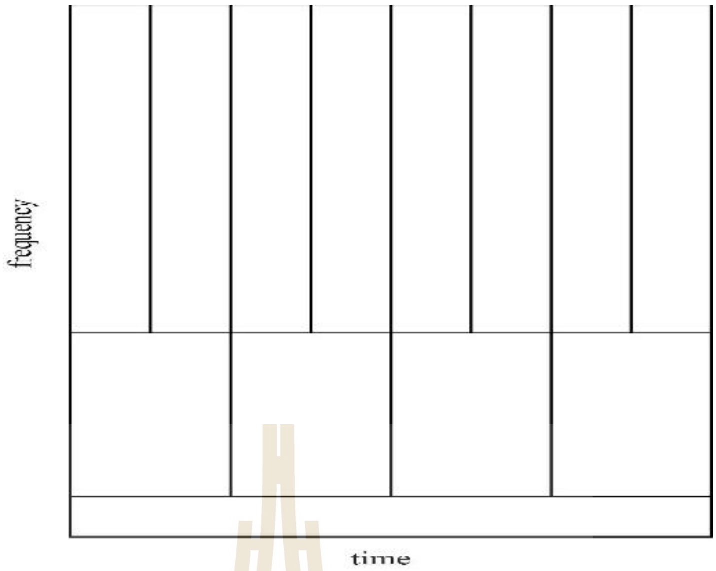

x[n] f=0~π
g[n]
h[n]
f=π/2~π
f=0~π/2
2
level-1
coefficients
g[n]
h[n]
f=π/4~π/2
f=0~π/4
2
level-2
coefficients


x[n]
split
even
odd
predict
update
cA1
cD1
split
even
odd
predict
update
cA2
cD2

Furthermore, LWT is a simpler way to implement DWT. The cost of LWT almost half less than DWT for LWT using lazy wavelet transform instead of filters (Daubechies & Sweldens, 1998). Hence, LWT is suitable for fall detection in respect of computational complexity.

2.2 Continuous Wavelet Transform (CWT)

Short-time Fourier transform (STFT) has been a widely used tool to analyze signals in time-frequency domain (Daubechies, 1992). STFT assumes that a non-stationary signal to be a stationary signal in narrow windows and does Fourier transform in this narrow window. However, there is an issue that first found by Heisenberg called "uncertain principle". The principle implies that we cannot know what frequency exists at what time instance but only know that what frequency bands exist at what time interval (Chui, 1992). In other words, if we know the exact location of a data point in time domain, we will never know the exact frequency of this data point at that time. As a result, there has to have a trade-off between time resolution and frequency resolution when choosing a constant window for STFT. A smaller window has a higher time resolution but lower frequency resolution and vice versa. Usually, ADLs are periodic and fall is a suddenly change of frequency. WT allows using long time window to obtain precise low-frequency information and short time window to localize high-frequency bands in time domain precisely. Let's take a closer look at this property of WT.



$$a,b) = \frac{1}{\sqrt{a}}\int_{+\infty}^{-\infty} f(t)\Psi \left(\frac{t-b}{a}\right)dt,$$

$t)$ i lled

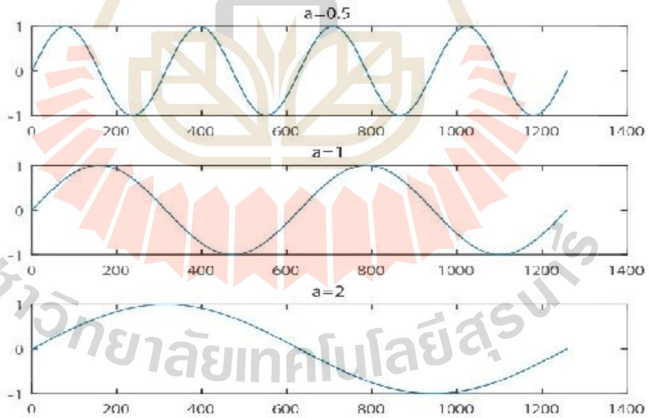
rtar $t)$ s

$t)$ i

$$\Psi(x) : \frac{x}{a} ,$$

$x \epsilon [\ \ a\pi, a\pi],\ and\ a \neq 0.$ Fo

$x)$



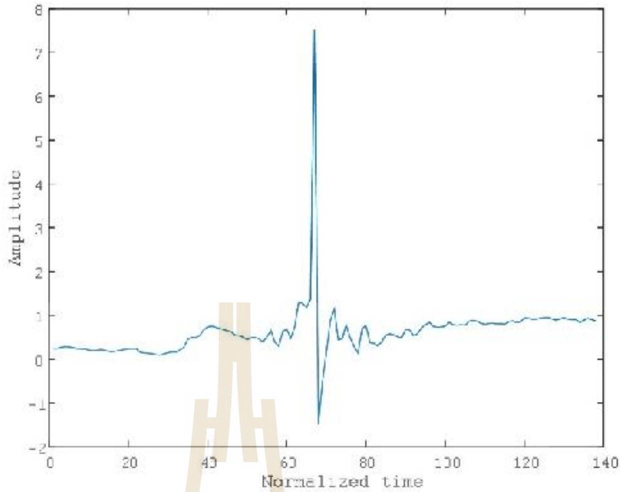
$\frac{x}{a}), a$

entire signal. Contrarily, lower scales correspond to less “stretched” (high frequency) wavelet and correspond to a hidden quick-changing details since sudden change usually lasts in a short time duration. Shifting or translation parameter “*b*” means “delay” or “advancing” the wavelet centered by value “*b*” in time-axis.

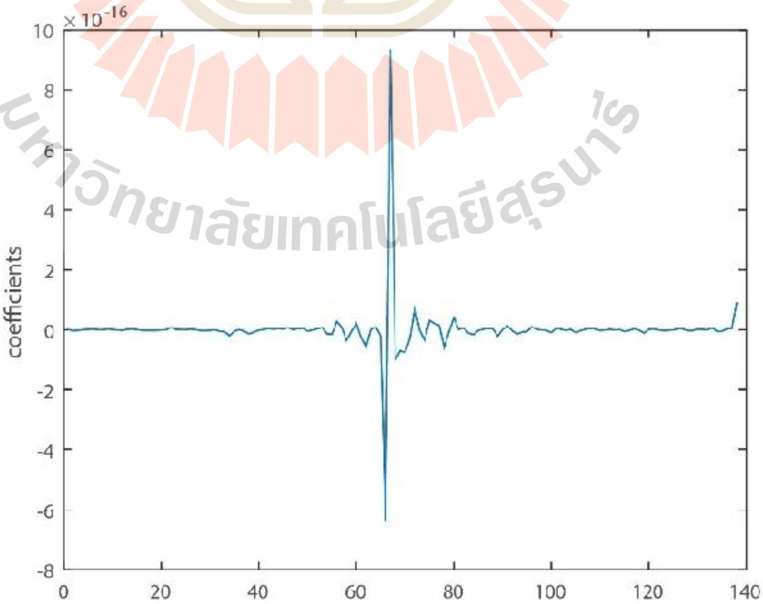
The definition of CWT given by formula (2. 2) is the inner product of signal function *f (t)* and transforming function$\Psi_{a,b}(t)$, where $\Psi_{a,b}(t)=\Psi\left(\frac{t-b}{a}\right)$. From this perspective, CWT definition shows the similarity between mother wavelet and the transformed function *f (t)* (Palmerini et al., 2015). The product value is non-zero only inside the support region of wavelet. It means that if the tested signal exists a spectral component correspond to current frequency (scale) and located inside current time interval (translation), the product value (CWT coefficients) will be relatively large. If no current spectral component exist or not located inside this time interval, the coefficients will be relatively small or even zero. Thus, by various scale and shifting values, wavelets in different location of time (interval) and frequency (bands) multiply with different part of tested signals.

CWT coefficients, on the other hand, becomes large around abrupt change in the signal since the abrupt transition in the shifted function results in large CWT coefficients at the discontinuity. As shown in

Figure 2.5 A typical fall with a clear peak, falls in daily life often come with a sudden change of frequency compared with periodic human activities. Thus, CWT adapt to detect a fall by observing the coefficient value of human activity signals. The CWT coefficients localize the discontinuity best at small scales for the reason that small support region of the wavelet ensures that the singularity only affects a small set of



Amplitude
Normalized time



coefficients

2.3 Support Vector Machine (SVM)

SVM is a popular machine learning method which was first proposed and proved by Cortes and Vapnik in 1995 (Cortes & Vapnik, 1995). It is a binary classification machine. The basic concept for SVM is that it is the largest margin classifier in the feature space. This is what makes SVM different from other machine learning methods.

The concept of SVM is to find the largest margin hyperplane to divide the negative and positive instances (samples). This hyperplane is used to predict instances whose labels are unknown. Moreover, if the instances are non-linearly separable, the use of some kernel function can map instances into a high dimension feature space where instances can be separated by a hyperplane.

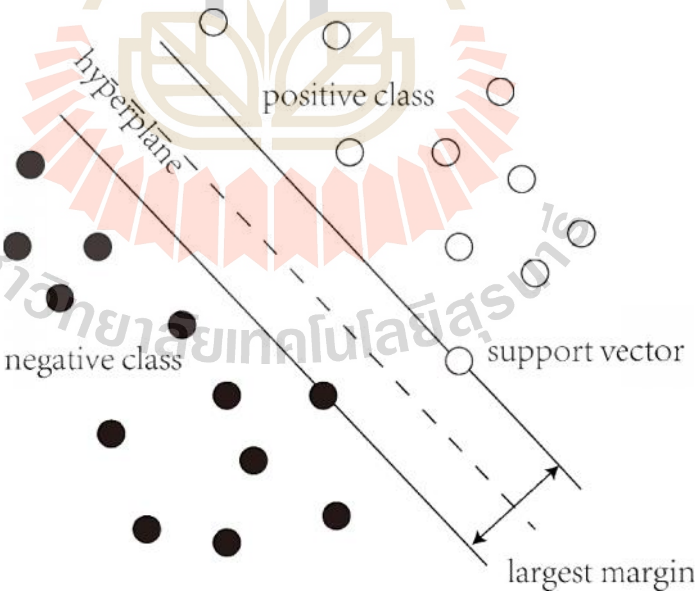


Figure 2.7 The largest margin depends on support vectors on the "edge"

Apart from kernel function, a soft margin method is also use to deal with non-linear problems. Thus, according to the concept of SVM, this section consists of three parts, the basic idea of SVM, the soft margin SVM and the kernel function, and the sequential minimal optimization (Scholkopf & Smola,2001).

2.3.1 Basic idea of SVM

Spouse that the input space and feature space are two different spaces. The input space is Euclidean space and the feature space is Hilbert space. The classifier assumes that the element in this two spaces correspond one-to-one. Assume that there exists a kernel function that can map the input space into a feature space. SVM classify in feature space. The input data set is called input instances. Assume that a group of input training data set $\mathrm{Tr} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_i, y_i), \ldots, (x_N, y_N)\}$, $1 \leq i \leq N$,where $\boldsymbol{x}_i \in \mathbb{R}^N$, is the i^{th} feature and y_i□{-1, +1} is the label of the i^{th} feature. When y_i equals +1, $\boldsymbol{x}_i$ is a positive sample vector and when y_i equals -1, $\boldsymbol{x}_i$ is a negative sample vector. Let $(\boldsymbol{x}_i, y_i)$ be the i^{th} instance. The objective of SVM is to find a hyperplane in feature space to classify the instance in feature space. A plane can be defined by a normal vector and an intercept. Therefore, if the training set Tr is linearly separable, the hyperplane can be defined by

$$\boldsymbol{\omega} \cdot \boldsymbol{x} + b = 0, \tag{2. 4}$$

where $\boldsymbol{\omega}$ is a normal vector of hyperplane and b is the intercept. Then, the decision is given by

$$f(\boldsymbol{x}) = \mathrm{sgn}(\boldsymbol{\omega} \cdot \boldsymbol{x} + b), \tag{2. 5}$$

where sgn is short for sign function. Equation (2. 5) implies that the classification result *f(x)* is compared with zero.

However, from

Figure **2.7**, it can located be seen that some instances are more important than the other because they are located at the edge of the margin. Those vectors are called support vectors. Apparently, there exists various hyperplanes (various $\boldsymbol{\omega}$ and *b*) to classify those instances. To determine a particular *b* for such classification, assume that the positive instance is larger than a constant. Similarly, we insist the negative instance is less than the opposite constant as well. For convenience, the constant is to be +1 and -1. In particular, the decision rule is defined by

$$\boldsymbol{\omega} \cdot \boldsymbol{x}_+ + b \geq 1, \quad (2.\ 6)$$

$$\boldsymbol{\omega} \cdot \boldsymbol{x}_- + b \leq -1, \quad (2.\ 7)$$

where $\boldsymbol{x}_+$ and $\boldsymbol{x}_-$ are positive support vector and negative support vector, respectively. Equations (2. 6) and (2. 7) can be rewritten in terms of y_i as

$$y_i(\boldsymbol{\omega} \cdot \boldsymbol{x}_i + b) \geq 1. \quad (2.\ 8)$$

The distance between hyperplane $(\boldsymbol{\omega}, b)$ and support vector is given by

$$d = (\boldsymbol{x}_+ - \boldsymbol{x}_-) \cdot \frac{\omega}{\|\omega\|}\ . \quad (2.\ 9)$$

By substituting $\boldsymbol{x}_+ \cdot \boldsymbol{\omega}$ and $\boldsymbol{x}_- \cdot \boldsymbol{\omega}$ by (2. 6) and (2. 7), we get

$$d = \frac{2}{\|\boldsymbol{\omega}\|} \; . \tag{2. 10}$$

Thus, the task of finding the largest margin between the two classes of instances becomes to find the maximum value of d, subject to (2. 8).

Now, we can construct an equivalent of such problem using

$$\min_{\omega,b} \frac{1}{2}\|\boldsymbol{\omega}\|^2, \tag{2. 11}$$

$$\text{s.t.} \;\; y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) \geq 1. \tag{2. 12}$$

The problem is proved feasible and has exactly one largest margin hyperplane. We are going to assume that the optimized result is hyperplane $(\boldsymbol{\omega}^*, b^*)$.

In order to solve the primal problem defined in (2. 11)-(2. 12), the dual problem is easier to solve. Let us now introduce the Lagrange function given by

$$\mathrm{L}(\boldsymbol{\omega}, \boldsymbol{\alpha}, b) = \frac{1}{2}\|\boldsymbol{\omega}\|^2 + \sum_{i=1}^{N} \alpha_i \, y_i \mathbf{0}\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) - \sum_{i=1}^{N} \alpha_i, \tag{2. 13}$$

where $= [\alpha_1, \alpha_2 \ldots, \alpha_i, \ldots, \alpha_N]^T$, $\alpha_i \geq 0$ is a Lagrange multiplier vector. The partial derivative of $\boldsymbol{\omega}$ and b equal to zero where the gradients of primal problem and constraint are parallel.

$$\nabla_{\omega} L(\boldsymbol{\omega}, \boldsymbol{\alpha}, b) = \boldsymbol{\omega} - \sum_{i=1}^{N} y_i \alpha_i \boldsymbol{x_i} = 0, \tag{2. 14}$$

$$\nabla_{b} L(\boldsymbol{\omega}, \boldsymbol{\alpha}, b) = \sum_{i=1}^{N} \alpha_i y_i = 0. \tag{2. 15}$$

Therefore,

$$\boldsymbol{\omega} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x_i}, \tag{2. 16}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0. \tag{2. 17}$$

Substitute into equation (2. 13), the dual problem is

$$\min_{\boldsymbol{\alpha}} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{x_i} \cdot \boldsymbol{x_j}) - \sum_{i=1}^{N} \alpha_i, \tag{2. 18}$$

$$s.t. \sum_{i=1}^{N} \alpha_i y_i = 0, \tag{2. 19}$$

$$\alpha_i \geq 0. \tag{2. 20}$$

Assuming that the optimize solution is $\boldsymbol{\alpha}^{*} = [\alpha_1^*, \alpha_2^*, \dots, \alpha_N^*]^T$. Thus, $\boldsymbol{\omega}^* = \sum_{i=1}^{N} \alpha_i^* \boldsymbol{x}_i y_i$. If chose one of $\alpha_j^* > 0$, is chosen such that $b^* = y_j - \sum_{i=1}^{N} \alpha_i^* y_i (\boldsymbol{x_i} \cdot \boldsymbol{x_j})$.

2.3.2 Soft Margin SVM and kernel function

In last section, we talked about the linear separable situation. However, the method is not available for non-linear training data unless a soft largest margin was proposed. More specifically, soft margin allows classifier make mistakes and add a penalty to the mistake.

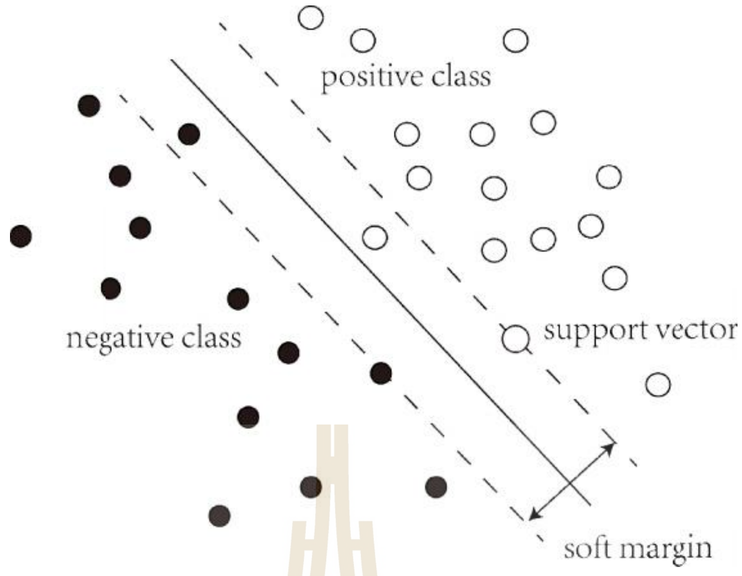


Figure 2.8 A soft margin SVM schematic

Similar process and hypothesis as previous section, we want the hyperplane to have largest margin as well as less mistakes. The soft margin SVM therefore can be described as

$$\min_{\omega,b,\xi} \ \frac{1}{2}\|\boldsymbol{\omega}\|^2 + C\sum_{i=1}^{N}\xi_i, \tag{2. 21}$$

$$\text{s.t.} \ \ y_i(\boldsymbol{\omega}\cdot\boldsymbol{x}_i + b) \geq 1 - \xi_i, \tag{2. 22}$$

$$\xi_i \geq 0, i = 1,2, \dots, \text{N}, \tag{2. 23}$$

where ξ_i represent scalar variables for the i^{th} instance, and C > 0 is the regularization parameter. Scalar variables are to characterize the unsatisfactory degree of this mistaken instance. And regularization is a classic way to control model complexity to avoid overfitting. Still, this is a convex quadratic problem as the “hard margin” SVM we talked in previous section. In hence, after Lagrange and duality, (2. 21)-(2. 23) are given by,

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(\boldsymbol{x_i}\cdot\boldsymbol{x_j}) - \sum_{i=1}^{N}\alpha_i, \tag{2. 24}$$

$$\text{s.t.}\quad \sum_{i=1}^{N}\alpha_i y_i = 0, \tag{2. 25}$$

$$0 \le \alpha_i \le C, i = 1,2,\dots,N. \tag{2. 26}$$

Compared (2. 24)-(2. 26) with (2. 18)-(2. 20), the only difference lays one the constraint (2. 20) and (2. 26).

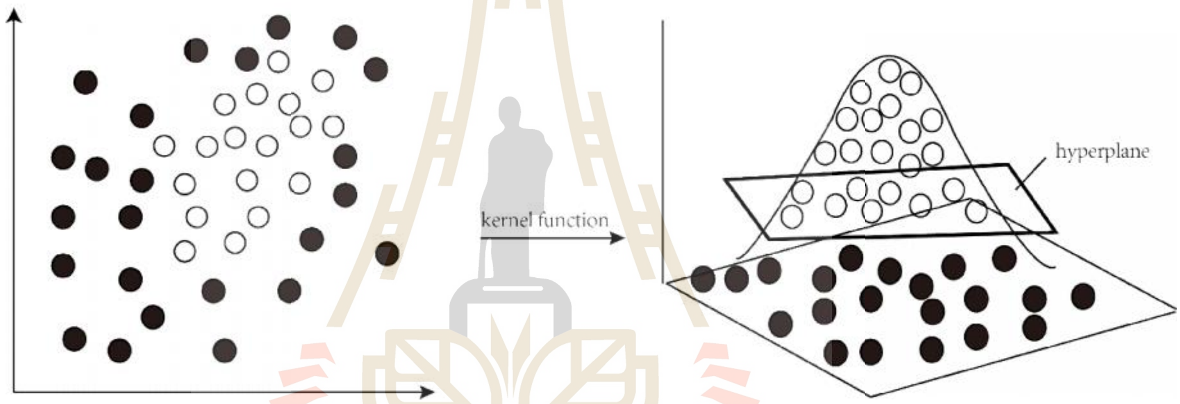


Figure 2.9 Kernel function example

Noticed that no matter in linear or non-linear situation, there only inner product instances $\boldsymbol{x_i}\cdot\boldsymbol{x_j}$ involved. For non-linear problems, for example in

Figure **2.7** and

Figure **2.9**, it is natural to think to map the data from low-dimension input space to high-dimension feature space that can find a hyperplane divide instances correctly. $\phi(\boldsymbol{x_i})$ is usually used to represent $\boldsymbol{x_i},\boldsymbol{x_j}$ in feature space, Thus, equation (2. 24) is given by

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y_i y_j \phi(\boldsymbol{x_i}) \cdot \phi(\boldsymbol{x_j}) - \sum_{i=1}^{N} \alpha_i, \tag{2. 27}$$

Unfortunately, the dimension of feature space can be high or even infinite, it is sometimes impossible to calculate directly in feature space. Due to the high dimensionality of feature space, researchers tried to define a kernel function,

$$\mathrm{K}(\boldsymbol{x_i}, \boldsymbol{x_j}) = \phi(\boldsymbol{x_i}) \cdot \phi(\boldsymbol{x_j}). \tag{2. 28}$$

Kernel function is an implicit function of inner product of feature space vectors. More information about the existence of kernel function, please refers to (Scholkopf & Smola, 2001). Substitute (2. 28) in to (2. 27),

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y_i y_j K(\boldsymbol{x_i}, \boldsymbol{x_j}) - \sum_{i=1}^{N} \alpha_i. \tag{2. 29}$$

Polynomial kernel function, Gaussian kernel function are often used in signal processing. For fall detection based on wearable sensors, radial basis function (RBF) classifier based on Gaussian kernel achieved best result (Hsu, Chang, & Lin, 2003). RBF is described as

$$\mathrm{K}(\boldsymbol{x}, \boldsymbol{z}) = \mathrm{e}^{-\frac{\|x-z\|^2}{2\sigma^2}}, \tag{2. 30}$$

where x and z are instances and σ is the various.

2.3.3 Sequential Minimal Optimization（SMO）and Cross Validation

Though the primal problem was transferred to dual problem which was proved to have and only have one optimize solution, it is still not easy to solve when the data size is large. This section, a method that is called sequential minimal optimization（SMO）will be introduced to solve dual problem.

SMO was first proposed by a scientist in Microsoft in 1998 (Chui, 1992). The method mainly focused on the convex quadratic programming problem looks like,

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x_i}, \boldsymbol{x_j}) - \sum_{i=1}^{N} \alpha_i, \quad (2.\ 31)$$

$$\text{s.t.} \quad \sum_{i=1}^{N} \alpha_i y_i = 0, \quad (2.\ 32)$$

$$0 \le \alpha_i \le C, i = 1,2, \dots, N, \quad (2.\ 33)$$

where N is the total number of data point.

SMO algorithm is a kind of heuristic algorithm. The idea of SMO is that the algorithm chooses a pair of Lagrange multiplayers (α_1, α_2) randomly as two unknown various and fix the left Lagrange multiplayers. From constraint (2. 32), we can get the relationship between this two parameters,

$$\alpha_1 = -y_1 \sum_{i=2}^{N} \alpha_i y_i, \quad (2.\ 34)$$

Thus, the N various problem becomes a two various sub problem with two equations. If we know one of the chosen unknown parameter, the other one is also fixed. In this term, the sub problem always updates two various together. The solution of sub problem will make the N various problem closer to final solution. Moreover, by transferring, the calculation speed is greatly improved. More information about how to solve a sub problem please look at (Chang & Lin, 2011).

There comes a problem that how to choose various pair. Firstly, we need to determine one of the two α. This is called outer loop. The instance that break the Karush-Kuhn-Tucker (KKT) conditions most seriously is chosen as the first various. In other words, the outer loop trends to choose support vectors as the first various for the reason that support vectors are at the edge of the hyperplane. Instances at this place are most likely to violate KKT conditions. If they satisfy KKT conditions, all the instance satisfy it.

Secondly, SMO uses inner loop to decide the next various. Assuming that in outer loop, the first various has been chosen. Inner loop trends to find the various that achieves largest change. If function $E(\mathbf{x}_i)$ represent the difference between the predicted y_i for input x_i and the real y_i, α that has larger $E(\mathbf{x}_i)$ is more likely to be chosen.

In conclusion, there are two main part for SMO algorithm. The first part is choose two various a time using heuristic method and the second part is to solve corresponding sub problem. Repeat this two parts, until all various satisfy KKT conditions.

The complexity of SMO depends on the number of support vectors instead of the number of feature space dimensions. This means SMO avoids to calculate in high-dimension space where overfitting trends to occur. Except this, cross validation helps to avoid overfitting as well.

As I mentioned above, SVM is a optimize problem, the result of this kind of machine learning method usually output various models in different complexities. In order to find a suitable model which is supposes to be the closest one to the best model. However, if we pursue high prediction accuracy of a model only, the chosen model is definitely more complex than “real model” which is assumed to be the perfect model.

This is called overfitting. There is an example of overfitting below.

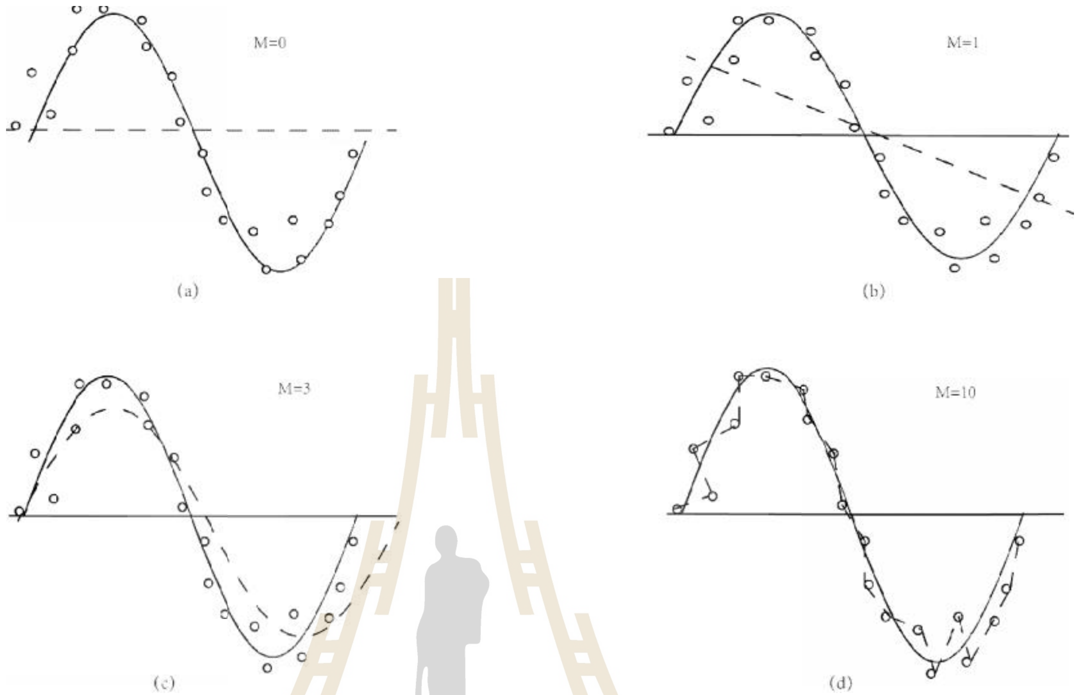


Figure 2.10 Overfitting

There are 20 data points in

Figure 2.10 Overfitting. We assume that the data set is represented by

$$\Omega = \{(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_{20}}, y_{20})\}, \tag{2. 35}$$

where $y_i = \{-1, 1\}$ is label of corresponding data. Then, if the data can be fitted by a polynomial given by

$$f(\boldsymbol{x}, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M = \sum_{i=0}^{M} w_i x^i, \tag{2. 36}$$

where w_i is parameters of this model. First of all, we need to know how many items we need to fit the given data. In other words, we need to decide the value of M and the complexity of the model.

Figure **2.10** shows different values of M. We can find that when M=0, the fitting curve is a constant which failed to match the sample data. When M= 1, the fitting curve is a line which also failed to match the sample perfectly. In contrast to this, when M=3, it looks that the fitting curve matches training data well enough. If the model is set more complex than this, for instance, M=10, the fitting curve matches training data perfectly and the deviation of model equals to zero for the training data set. It seems to be the best model for training data. However, generalization ability which represents the ability of model to predict unknown data is low. The reason overfitting occurs is that training data size is limited while the unknown data various.

Cross validation is another way to choose models. In this thesis, a 5-fold cross validation was used. The simplest cross validation will be illustrated below.

At the beginning, given dataset is divided into two parts according to a specific ratio randomly. One part is used as training dataset and the other part is used as testing dataset. Using training dataset in different situations and settings, models with different parameters are obtained. Those models are evaluated by the testing dataset, and the model with least deviation is chosen.

The cross validation we often use is S-fold cross validation. Instead of divide dataset into two part, S-fold divide dataset into S subsets equally. And then, S-1 subsets are used as training dataset and the left one is used for testing. Repeat this S times until all the subset has been used as testing set. Finally, the model with least average testing error is the model we want to find (Kohavi, 1995).

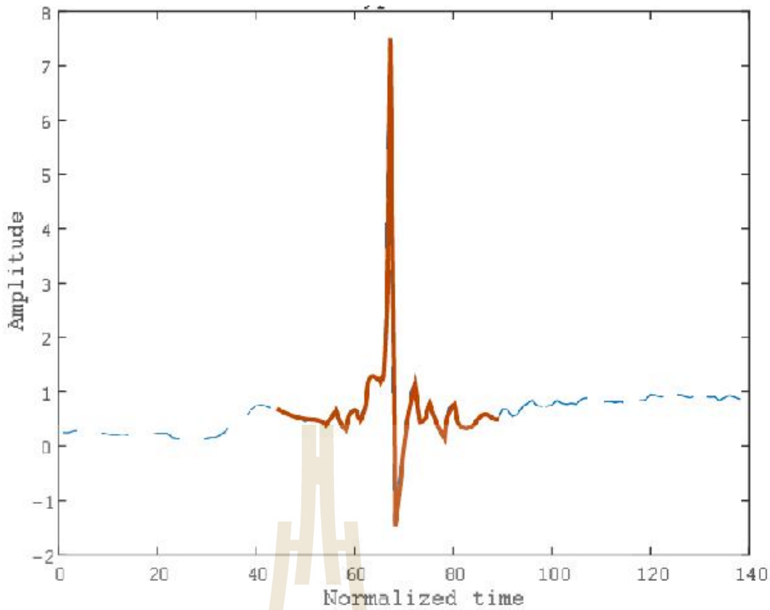
2.4 Others

Datasets used in this thesis come from (Kwolek & Kepski, 2014) and Imperial of London, respectively. Both of the data are collected by a tri-accelerometer attached to the waist of objects.

The dataset used in chapter III of this thesis is the former one. Those activities were conducted by 14 young volunteers. Two kinds of falls (from standing to fall and from sitting to fall) and four kinds of ADLs (squatting, sitting, lying down and bend over) were conducted. The dataset used in chapter IV is the latter one. There were thirteen kinds of falls and eleven kinds of ADLs.

2.4.1 Feature extraction

For the reason that fall detection data are collected in a long term and the sample rate is about 60 Hz which is relatively high. There is a large volume of data need to be processed in a short period of time. On the other hand, the data is not always equally important in terms of human activity recognition. Thus, before applying SVM on collected data, feature extraction is necessary. Larger dataset leads to more time to calculate. Before input this data into SVM which is relatively complex, windowing and labeling are used to prepare the data as well.



$${}_{\text{total}} = \sqrt{{A_x}^2 + {A_y}^2 + {A_z}^2},$$

where A_x, A_y, and A_z are the value of correspond oriental.

2.4.2 Threshold

This thesis compared the performance of different features including CWT and LWT coefficients. And compared the result of SVM with and without threshold. Threshold in chapter III was used to decide a fall according to the ratio of data point labelled as “+1” in an activity. In chapter IV, the threshold was calculated before wavelet transform to select falls which were difficult to distinguish from ADLs. It means that if the maximum SV_{total} of an activity was larger than threshold value, this activity would be regard as a fall directly. Detail algorithm would be illustrated in later chapters.

2.5 Summary

In this chapter, background theory of the algorithm proposed in this thesis were talked. Support Vector Machine (SVM) is a supervised machine learning method which has been widely used in data classification and regression analysis. The classifier tries to divide the falls and activities of daily living (ADLs) into two parts correctly. Basic SVM is good at dealing with linear problems which human activities data collected by wearable sensor is usually not. Thus, kernel function is introduced to map the data from input space to higher dimension feature space. Then a hyperplane (model) based on training dataset can be found in feature space. In order to avoid overfitting which is a common problem for modeling with limited training data, soft margin and cross validation were described. Mathematical definition and explanation were given following the concept. In particularly, SVM was attributed to a convex quadratic programming problem which sequential minimal optimization (Scholkopf & Smola, 2001) is one of the most efficient methods to solve.

After that, continuous wavelet transform (CWT) and lifting wavelet transform (LWT) were illustrated in the following two sections. Both CWT and LWT are included in wavelet transform (WT). CWT insists the scale and shift change continuously while that of LWT is discrete. From the definition, CWT coefficients can be regard as the similarity degree between mother wavelet and input signals. Therefore, an average fall was used to draw the similarity as a frequency domain feature for SVM.

LWT is an alternative of discrete wavelet transform (DWT). Instead of using filters which is complicated to split signals, LWT simply spilt input sequence into even and odd. LWT has advantages over CWT in computational complexity as well as better performance we suppose.

What last but not least was that the raw data which come from wearable sensors cannot be used directly since the data has three orients for one sample. Pre-processing helps to decrease the volume of data without losing useful information. The methods and equations about how to processing raw data before they are input in to wavelet transform algorithm and SVM are briefly described in the last section of this chapter. The detail illustration can be found on later chapters.

CHAPTER III

FALL DETECTION USING LIFTING WAVELET TRANSFORM AND SUPPORT VECTOR MACHINE

Frequency domain features of inertial movement enables multi-resolution analysis for fall detection, yet they are computationally intensive. This chapter proposes a computationally light frequency domain feature extraction method based on lifting wavelet transform (LWT) which provides computational efficiency suitable for real-time low power devices such as wearable sensors for human fall detection. LWT is combined with support vector machine (SVM) to identify falls from activities of daily living. Performance of the Haar and Biorthogonal 2.2 wavelets were compared with the time domain feature of root-mean square acceleration using a human fall dataset. Results show that the first level detail coefficients features for both Haar and Biorthogonal 2.2 wavelets achieved good overall accuracy, sensitivity and specificity.

3.1 Introduction

As many countries enter the era of aging society, they face critical elderly people's health threats which are fall and related complications caused by the injury (Pierleoni, Pernini, et al., 2015). Considering the need of real-time monitoring and ease of use, wearable sensor systems are one of the most promising systems.

Wearable sensor-based fall detection systems, inherently generate continuous monitoring of physiological measurements. Such system is usually a multi-sensor system. Comprising sensors such as accelerometers, gyroscopes, pressure sensors and magnetometers. Datasets collected by such wearable sensors are thus, typically multi-dimensional and in large volumes. Such characteristics may cause hinder data processing and fall detection capabilities. Some researches therefore use feature extraction to reduce the amount and the dimensions of data (Banaee et al., 2013) by extracting only necessary features. Existing techniques include two main domains, i.e., time and frequency domains. Research such as (Pierleoni, Pernini, et al., 2015), (Carlsson, 2015), (Özdemir & Barshan, 2014) extracted time domain features including the mean value, maximum value, minimum value and variance, standard deviation of the patient's physiological movements and other special features such as entropy and vertical direction.

In general, time domain features are straightforward and easy to visualize which means light computational burden for feature extraction. So the system is computationally efficient in achieving a real-time fall detection. However, the time domain statistical features considers only the displayed observable trends (Özdemir & Barshan, 2014). Consequently, time domain features may not suffice for accurate fall detection.

Conversely, frequency domain features make use the spectral domain of the collected data which may not be clearly observable in the time domain. Frequency domain features were deployed for fall detection by (Su, Ho, Rantz, & Skubic, 2015) which used discrete stationary wavelet transform (SWT). In (Björklund et al., 2015), a short time Fourier transform (STFT) was used for human activity recognition, whereby a fall was a subset of data in a series of continuous activities of daily living (ADLs). In (Palmerini et al., 2015), created a prototype wavelet of typical fall pattern by using the average acceleration sum vector. The degree of similarity of the signal to the prototype was then computed though wavelet analysis. Results from the same classifier and real-world dataset revealed that the wavelet based features outperformed than other time domain features: upper and lower peak values.

Feature extraction alone only enhance the features of the data acquired by the wearable sensors. However, to detect weather a fall occurred relies on the performance of the detection mechanism. The most common and simplest fall detection is the threshold method (Aziz et al., 2017). Nevertheless, the performance heavily depends on the fixed threshold level. Hence, it is rarely used alone, and often combined with other machine learning methods such as decision tree (DT) (Bilski, Mazurek, Wagner, & Winiecki, 2015), (Parkka et al., 2006), artificial neural networks (ANN) (Z. Wang et al., 2012), hidden Markov model (HMM) (Tong et al., 2013) and Support Vector Machine (SVM) (Özdemir & Barshan, 2014), (Pierleoni, Belli, et al., 2015), (Liu & Cheng, 2012) can be combined to outperform the threshold method (Aziz et al., 2017), (Aziz et al., 2017). Among the machine learning methods, SVM was found the most robust for fall detection when compared to other methods such as threshold-based methods and the decision tree method (Aziz et al., 2017). However, most works which deploy SVM for fall detection use time-series features (Aziz et al., 2017), (Shibuya et al., 2015). It was found that SVM fall detection performance can be improved by a combination of time and frequency domain features (Özdemir & Barshan, 2014). In particular, the discrete Fourier transform (DFT) was used to determine the spectral coefficients which is computationally intensive (Özdemir & Barshan, 2014). On the other hand, the lifting wavelet transform (LWT) is an efficient, light weight frequency domain extraction method (Sweldens, 1998). To the best of our knowledge, there is no previous work that has combined LWT with SVM for fall detection. This chapter is therefore focused on the study of feature extraction based on LWT used with SVM to detect falls from ADLs using root-mean square value from a single tri-axial acceleration sensor.

The part is organized as follows. Section II presents the proposed frequency analysis and the support vector machine scheme proposed in the chapter. The time domain feature which is used for comparison is also introduced. In section III, the experiment based on a comprehensive fall detection dataset is described. Section IV presents the results and discussion and finally conclusions is given in the final section.

3.2 Method

3.2.1 Frequency domain feature extraction

Feature extraction based on frequency analysis of the body inertia collected from sensors has been studied in the recent literature. Discrete wavelet transform (DWT) has been proposed for mobility monitoring, posture transition and activities classification in (Wójtowicz, Dobrowolski, & Tomczykiewicz, 2015b) using a single chest-mounted sensors. In (Shin et al., 2015), another frequency domain feature extraction method using short-time Fourier transform (STFT) was proposed to shorten the calculation time of DWT. Despite good results, the short time windows in STFT may not always be suitable for human motion which varies greatly. If windows are too short, STFT may be unable to identify the frequency in such a short period of time. If windows are too large, more information in time domain will lost. If the STFT window size is fixed, STFT may not be suitable for fall detection as human activities are flexible. Unlike DFT in (Özdemir & Barshan, 2014), LWT can be constructed from time series signal directly. Unlike DWT in (Wójtowicz et al., 2015b), LWT does not require convolution, translation or dilation of traditional mother wavelets. Furthermore, LWT allows in place calculation, with no need for auxiliary memory. Therefore, LWT provides computational efficiency suitable for real-time low power devices such as wearable sensors. In the following subsection, we describe LWT in more details.

3.2.2 Lifting Wavelet Transform

LWT has been introduced by Sweldens in 1997 (Sweldens, 1998). The scheme theory is often described as three steps: split, predict and update. The split step is to split a signal into to two independent sequences, i.e, the even half and odd half sequences. Let x_i be the original discrete signal at time index *i*. Let $even_i$ (odd_i) denote the i^{th} index of the even (odd) sequence. We have that $even_i = x_{2i}$ and $odd_i = x_{2i+1}, i \in I$.

LWT is a recursive algorithm whereby if the original signal has 2^n elements, then the next level will operate on $2^{(n-1)}$ elements. Hence, if the original signal has 256 elements, there will be 8 levels with the next level having 128 elements. The subsequent levels will have 64, 32, 16, 8, 4, 2 and 1 element. The odd values in the next level *j+1* is predicted from the even value at level *j*:

$$cD_{j+1,i} = odd_{j,i} - P(even_{j,i}), \tag{3. 1}$$

where *P* is the predict function which approximates the signal. And cD is the high frequency part of x_i used to replace $odd_{j,i}$. This is called the *Predict* phase. The even values at the next level can be found from

$$cA_{j+1,i} = even_{j,i} - U(cD_{j+1,i}), \tag{3. 2}$$

where *U* is the update operation that updates on the differences from the odd values. And cA is the low frequency part of x_i used to replace $even_{j,i}$. This is called the *Update* phase. The multi-level lifting scheme can be summarized in Figure **3.1**. The averages are sometimes called approximate coefficients whereas the differences are called the detail coefficients. There are two types of wavelets used in this chapter.

1) Haar wavelet:

Predict:

$$cD_{j+1,i} = odd_{j,i} - even_{j,i}. \tag{3. 3}$$

Update :

$$cA_{j+1,i} = even_{j,i} + \frac{1}{2} cD_{j+1,i}. \tag{3. 4}$$

2) Biorthogonal 2.2 wavelet:

Predict :

$$cD_{j+1,i} = odd_{j,i} - \frac{1}{2}(even_{j,i} + even_{j,i+1}). \tag{3. 5}$$

Update :

$$cA_{j+1,i} = even_{j,i} + \frac{1}{4}(cD_{j+1,i-1} + cD_{j+1,i}). \tag{3. 6}$$

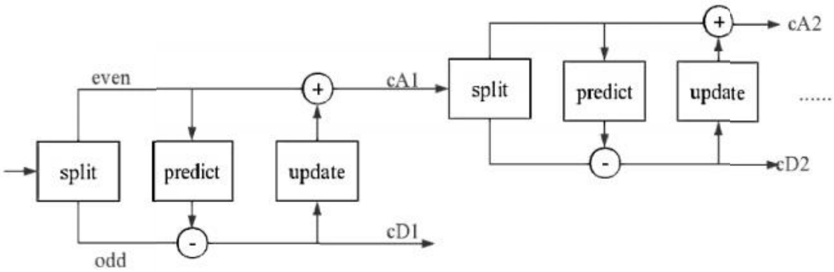


Figure 3.1 Forward lifting scheme

Figure **3.2** shows a sample fall plot of the original signal and the *first* level LWT coefficient. The number of coefficients of cA1 (average or low frequency part) and cD1 (detail or high frequency part) are half of the original signal according to the number of data points. By comparing cA1, cD1 and the root-mean square acceleration (SV_{total}) in Figure **3.2**, it is seen that cA1 greatly correlates with the original signal. Note that cD1 also shows a peak similar to the original signal signifying a fall which occurred during the red highlighted window of one second. However, the baseline zero illustrated a more distinguished fall feature than cA1. Therefore, cD1 was preferable than cA1 for feature extraction of falls.

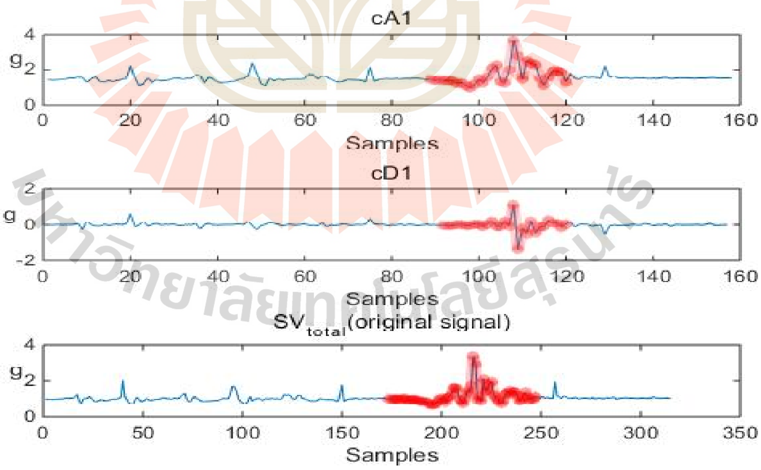


Figure 3.2 A sample fall plot of SV_{total} and after LWT cD1 and cA1 with data points inside window highlighted

3.2.3 Time domain feature

The tri-axial acceleration data collected contains A_x, A_z, A_y in x-axis, z-axis and y-axis as a function of time. All accelerometer data were in factors of gravity units (g). The accelerometer components were used to calculate the root-mean square acceleration denoted by total sum vector SV_{total}:

$$SV_{total} = \sqrt{{A_x}^2 + {A_y}^2 + {A_z}^2}. \quad (3.7)$$

3.2.4 Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning model which is commonly used for anomaly detection and classification (Hsu et al., 2003), (Cortes & Vapnik, 1995), (Chang & Lin, 2011). As a supervised learning model, SVM requires training from datasets with “labels.” The SVM concept is to map a set of data points from the real-world to a higher dimensional space. A boundary or hyperplane is created in a high dimensional space by training datasets to classify the features into fall or non-fall. Since the fall detection system inherently generates long-term continuous monitoring of physiological measurements, such datasets are usually large. Such characteristic may cause difficulty in data processing. To reduce the amount of data and achieve a higher calculating speed, the features of the data may be extracted from these raw datasets.

To train the SVM, the data points in the dataset must be labeled. For example, in time domain, SV_{total} was directly used as input feature. We labeled all the ADLs data points with “-1” whereas falls were labeled “1.”

Figure **3.3** depicts a sample plot of a fall along with non-fall activities like walking around and lying on the ground. Point A shows the peak value of the dataset.

A highlighted window size with point A placed at the middle of the window is constructed. Within such window, all the data points are labeled “1” and the remaining data points outside this window are labeled “-1.” The goal of SVM is therefore to distinguish the labels among the tested datasets using the model obtained from the trained data. The data points are typically non-linearly separable to classify in low dimensional space. However, if these points are projected onto a higher dimensional space, it is possible to find a hyperplane to classify the labels. Such projection is obtained through use of kernel functions such as linear, polynomial, sigmoidal, or the Gaussian radial base functions. It is with this kernel trick that makes SVM a powerful model to classify the labels in higher dimensional space. In the next section, the experiment settings are presented.

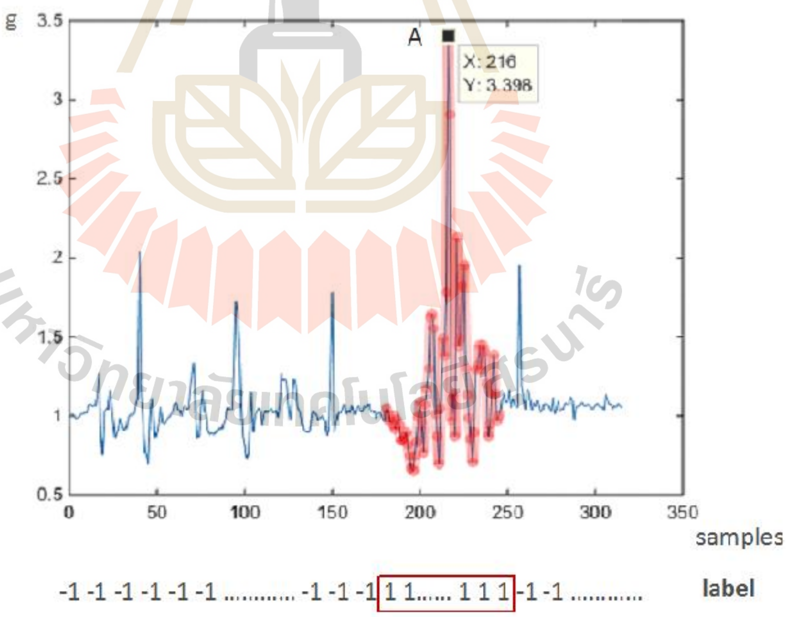


Figure 3.3 A sample plot of fall SV_{total} with data points inside widow highlighted

3.3 Experiment

As mentioned in the previous section, SVM requires training labeled datasets.

As data input in the fall detection scenario involves both non-falls and falls data, we trained with both falls and non-falls italic in Table 3.1.

We first evaluate different SVM model with three different groups of activities, namely, non-fall only activities, fall only activities and a hybrid fall and non-fall activities. The objective is to determine the suitable training dataset for SVM model to detect falls. For the sake of simplicity, only the time domain feature (SV_{total}) is studied.

Once a suitable SVM model is found, we proceed to study the comparison between features in the time domain and frequency domain. Note that there are existing works which combined features in both time domain and frequency domain of data, the type of sensors, the number and position of sensors on human body, and in the volume of dataset for training and testing (Pierleoni, Pernini, et al., 2015), (Su et al., 2015). From results gathered from existing literature, we focus on data collected from a single tri-axial acceleration sensor due to its low cost, reliability and efficiency.

3.3.1 Performance metrics

We measure the True Positives or True Negatives which refer to the number of events correctly identified or correctly. Fall Detection using Lifting Wavelet Transform and Support Vector Machine. It is worth noting that SVM classifies data points individually. However, to detect a fall within a certain window as shown in Figure **3.3**, a set of data points must be classified rather than just a single data point.

Therefore, to determine a suitable decision region to decide whether a fall has occurred, we use a simple calculation for the percentage of predicted fall label “1” over the number of labels observed in an activity to compare with a fixed threshold:

$$Th = \frac{the\ number\ of\ predicted\ "1"}{the\ number\ of\ testing\ data\ points}. \tag{3. 8}$$

If *Th* > threshold, the activity is a fall, else it is non-fall.

Table 3.1 Datasets used in this experiment[1]

Data File	Activities Description	Data File	Activities Description
		Falls	
fall-01-acc	*From vertical falling left on the floor*	**fall-16-acc**	From sitting falling right on the floor
fall-02-acc	*From sitting falling left on the floor*	**fall-17-acc**	From vertical falling forward on the floor
fall-03-acc	From vertical falling left on the floor	**fall-18-acc**	From sitting falling left on the floor
fall-04-acc	From sitting falling left on the floor	**fall-19-acc**	From vertical falling right on the floor
fall-05-acc	*From vertical falling right on the floor*	**fall-20-acc**	From sitting falling right on the floor
fall-06-acc	*From sitting falling right on the floor*	**fall-21-acc**	From vertical falling right on the floor
fall-07-acc	From vertical falling left on the floor	**fall-22-acc**	From sitting falling left on the floor
fall-08-acc	From sitting falling right on the floor	**fall-23-acc**	From vertical falling right on the floor
fall-09-acc	From vertical falling left on the floor	**fall-24-acc**	From sitting falling left on the floor
fall-10-acc	From sitting falling left on the floor	**fall-25-acc**	From vertical falling forward on the floor
fall-11-acc	From vertical falling right on the floor	**fall-26-acc**	*From sitting falling forward on the floor*
fall-12-acc	From sitting falling right on the floor	**fall-27-acc**	From vertical falling forward on the floor
fall-13-acc	*From vertical falling forward on the floor*	**fall-28-acc**	From sitting falling forward on the floor
fall-14-acc	From sitting falling right on the floor	**fall-29-acc**	From vertical falling forward on the floor
		Non-falls Activities (ADLs)	
Data File	**Activities Description**	**Data File**	**Activities Description**
adl-01-acc	*Walking, then squatting*	**adl-21-acc**	From vertical lying on the bed
adl-02-acc	Walking, then squatting	**adl-22-acc**	From vertical lying on the bed
adl-03-acc	Walking, then squatting	**adl-23-acc**	From vertical lying on the bed
adl-04-acc	*Bending 90 degree to pick up something*	**adl-24-acc**	Walking, then squatting
adl-05-acc	Squatting to pick up something	**adl-25-acc**	From vertical to sitting onto a chair
adl-06-acc	Squatting to pick up something	**adl-26-acc**	Walking, then squatting
adl-07-acc	*From vertical to sitting onto a chair*	**adl-27-acc**	From vertical to sitting onto a chair
adl-08-acc	From vertical to sitting onto a chair	**adl-28-acc**	Walking, then squatting
adl-09-acc	From vertical to sitting onto a bed	**adl-29-acc**	From vertical to sitting onto a chair
adl-10-acc	*From vertical lying on the bed*	**adl-30-acc**	From vertical lying leftward on the ground
adl-11-acc	From vertical lying rightward on the bed	**adl-31-acc**	*From vertical lying forward on the ground*
adl-12-acc	Walking, then squatting	**adl-32-acc**	From vertical lying forward on the ground
adl-13-acc	Walking, then squatting	**adl-33-acc**	From vertical lying forward on the ground
adl-14-acc	Walking, then squatting	**adl-34-acc**	From vertical lying forward on the ground
adl-15-acc	Bending 90 degree to pick up something	**adl-35-acc**	From vertical lying forward on the ground
adl-16-acc	Bending 90 degree to pick up something	**adl-36-acc**	From vertical lying rightward on the ground
adl-17-acc	*Squatting to pick up something*	**adl-37-acc**	From vertical lying rightward on the ground
adl-18-acc	From vertical to sitting onto a bed	**adl-38-acc**	From vertical lying forward on the ground
adl-19-acc	From vertical to sitting onto a chair	**adl-39-acc**	From vertical lying forward on the ground
adl-20-acc	From vertical to sitting onto a bed	**adl-40-acc**	From vertical lying forward on the ground

3.3.2 Evaluating SVM Model

We hypothesized that the best type of training dataset will be the combined set of both fall and ADLs dataset. Since not only falls but also ADLs data are contained in the hybrid training dataset, the more comprehensive information contained

[1] The italic activities were used as training data set

in training dataset, the more correctly decision the model will make.

The dataset we used to train and test the SVM models were from (Kwolek & Kepski, 2014) including 70 activities (tri-axial acceleration of 30 falls and 40 non-falls collected and video recorded with Kinect camera) with details given in Table 3.1.

The tri-axial accelerometer data was sampled at 60Hz. Therefore, a one-second window for fall detection consists of 60 data points. The dataset was divided into training set and testing set based on activities in the matching video of each data file. Table 3.1 consists of fall and non-fall (ADLs) activities. For simplicity, only the hybrid-dataset-training model is used to evaluate the performance of features. The models under study include:

Model-1 (ADLs only). To learn a wide variety of non-fall activities, the following datasets were used to train model-1, including, adl-01-acc, adl-04-acc, adls-07-acc, adl-10-acc, adl-31-acc were chosen as training dataset.

Model-2 (Falls only). To train the falls only model we used a wide variety of fall datasets, including, fall-01-acc, fall-02-acc, fall-05-acc, fall-06-acc, fall-13-acc, fall-26-acc.

Model-3 (trained by Falls and ADLs). This SVM model was trained with all datasets previously used in model-1 (ADLs only) and model-2 (Falls only).

Once the data points are labeled and trained, SVM models based on the training dataset are obtained. The SVM models are then used to classify the testing data. The dataset remaining (not italic activities) in Table 1 is used for testing. For each dataset tested, a data point is labeled “1” for data points predicted as a fall data point, or “-1” for data points predicted as non-fall data point. If the ratio of fall labels in an activity exceeds the determined threshold, then a fall has been detected. For each tested dataset, TP, TN, FP and FN is measured for the calculation of SE, SP and AC to evaluate the SVM models. Results are shown in Table 3.2.

Table 3.2 SVM model comparison[2] for time domain feature

Training Data	Model-1 (ADLs)	Model-2 (Falls)	Model-3 (Both)
SE (%)	100	100	**100**
SP (%)	0	35	**97.14**
AC (%)	46.88	59.32	**98.31**

3.3.3 Comparing Time and Frequency domain feature

This part of the experiment is to compare the time domain feature (based on SV_{total}) and the frequency domain features (based on Haar and Biorthogonal 2.2 wavelets). Using the SVM models obtained in the previous experiment, a suitable level threshold level to detect a fall event for each feature is then found. For each feature, the percentage levels of threshold is tested at 10%, 20%, 30%, 40% and 50%. Then level is tested at finer threshold values. Results are shown in Table 3.3.

Table 3.3 Performance comparison[3] at different thresholds of time and frequency domain features

Systematics Metrics	Initial Estimate Threshold	Fine Tuned Threshold
	Time Domain (SV_{total})	

[2] Bold fonts indicate the best performance

[3] Trained with SVM Model-3 and tested by ADLs and Falls dataset

Threshold	10%	20%	30%	40%	50%	15%	**17%**[4]	**18%**	19%
SE (%)	100	95.83	91.67	87.50	87.50	100	**100**	**100**	100
SP (%)	80	100	100	100	100	94.29	**97.14**	**97.14**	91.43
AC (%)	87.93	98.28	96.55	94.83	94.83	96.55	**98.28**	**98.28**	94.83
	Frequency Domain (Haar, cD1)								
Threshold	10%	20%	30%	40%	50%	2%	4%	6%	**8%**
SE (%)	95.83	87.50	87.50	66.67	33.33	100	100	100	**100**
SP (%)	100	100	100	100	100	82.35	85.29	97.06	**100**
AC (%)	98.28	94.83	94.83	86.21	72.41	89.66	91.38	98.28	**100**
	Frequency Domain (Bior 2.2, cD1)								
Threshold	10%	20%	30%	40%	50%	4%	5%	**6%**	
SE (%)	95.83	87.50	83.33	62.50	33.33	100	100	**100**	
SP (%)	100	100	100	100	100	88.57	97.14	**100**	
AC (%)	98.31	94.92	93.22	84.75	72.88	93.22	98.31	**100**	

We then investigate closely how multiple levels of LWT coefficients affect the fall detection performance by evaluating the first five levels of coefficients of the Haar and Biorthogonal 2.2 wavelets. Only the SVM model which performed the best from the previous experiment was evaluated. Results are shown in Table 3.3 and Table 3.4.

Table 3.4 Performance comparison[5] at different components[6] in frequency domain

Features / Metrics	**cD1**	**cD2**	**cD3**	**cD4**	**cD5**
	Haar				
SE (%)	100	83.33	95.83	87.50	95.83
SP (%)	100	100	100	100	100
AC (%)	100	93.10	98.28	94.83	98.28
	Bior 2.2				
SE (%)	100	100	91.67	100	100
SP (%)	100	94.29	100	82.86	5.71
AC (%)	100	96.61	96.61	89.83	44.07

3.4 Results and Discussion

3.4.1 Results and Discussion of Evaluating SVM Models

The experiment shows the sensitivity, specificity and accuracy of the

[4] Bold fonts indicate the best performance for each feature
[5] Trained with SVM Model- and tested by ADLs & Falls dataset
[6] Bold fonts indicate the best performance for each feature

SVM model. The results show that model-1 trained by ADLs only performed the worst because no ADL recognized from a ADLs-only testing set. Model-3 the best sensitivity, specificity and accuracy is 100%, 97.14%, 98.31%, respectively. This 100% of sensitivity means all falls were detected. It may be because model-1 only had no-fall label of “-1,” and fall label “1” was not used in the one-class model in LIBSVM. The falls in the dataset collected from a mere handful of people, not as many patterns as in real world.

3.4.2 Results and Discussion of Comparing Time and Frequency domain feature

Table 3.2 shows the performance comparison between time and frequency domain features at different levels of thresholds.

Root-mean square acceleration: Table 3.2 shows that the best threshold for the time domain feature should be between 10% to 20%. With fine threshold tuning, it is found that a threshold of 17-18% showed better preference than others (shown in bold fonts). Therefore, we chose 17% as the threshold to classify a fall or non-fall for time domain features.

LWT with Haar Wavelet: The appropriate threshold for Haar LWT is found by also ranged from 10% to 50%. As shown in Table 3.2, the best achieved threshold should be under 10%. To fine tune the threshold levels, the threshold is varied from 2% to 10%. It is found that the threshold at 8% outperformed other levels (shown in bold fonts). Thus, we chose 8% as the threshold for LWT using Haar wavelet. In Table 3.3, multiple levels of LWT coefficients (cD1 to cD5) are evaluated. When tested with ADLs & Falls dataset, all specificity, specificity and accuracy values of 100% was achieved only in cD1 (shown in bold fonts). This result indicated that Haar LWT CD1 coefficients achieved a goal that no ADL was misclassified as a fall and detected most of the falls when training and testing using finite activities in Table 3.1.

LWT with Biorthogonal 2.2 Wavelet: From Table 3.2, the optimal threshold for biorthogonal 2.2 (Bior 2.2) should be under 10% as well. With a finer threshold search, results indicate that threshold level of 6% is the best level with 100% sensitivity, specificity and accuracy (shown in bold fonts). Similar to Haar LWT, Bior2.2 LWT coefficients also show a good performance distinguishing falls from ADLs when using most cD levels. In Table 3.3, cD1 also outperformed other levels of coefficients similar to Haar wavelet (shown in bold fonts). The reason maybe the information contained by the data that is helpful when using SVM to classify activity. Such information is level by level. Thus, cD1 had the most information while cD5 had the least information. Generally, Haar was slightly better at distinguish ADLs from falls than Bior2.2, whereas both LWT features outperform the root-mean square acceleration alone. It is worth noting that these results are obtained by a comprehensive human fall dataset with video captures obtained from (Kwolek & Kepski, 2014) which allow the thresholds and detail coefficients can be predetermined. Current ongoing work involves implementing the LWT and SVM on actual wearable sensor devices to be evaluated online for human fall detection for accuracy and efficiency.

3.5 Summary

In this chapter, we propose a computationally light frequency domain feature extraction method called lifting wavelet transform (LWT) for a wearable sensor human fall detection device combined with a fall identifier using support vector machine model. The performance of the LWT using Haar and Biorthogonal2.2 wavelets, together with the time domain feature of root-mean square acceleration have been evaluated with raw dataset acquired from a single tri-axial acceleration sensor from an existing human fall and activities of daily living dataset.

Based on the dataset, suitable thresholds and level of detail coefficients can be predetermined. Consequently, the LWT frequency domain features are shown to have better performance than time domain features in terms of sensitivity, specificity and accuracy. Given a one-second window size under a sampling frequency of 60Hz, the best threshold in terms of the percentage of fall labels “1” per window is as follows, 18% for the time domain feature using the root-mean square acceleration, 8% for Haar and 6% for Biorthogonal2.2 LWT wavelets when the SVM model is trained with both fall and non-fall datasets (Model-3). The frequency domain feature from cD1 for both Haar and Biorthogonal2.2 wavelets achieved 100% overall accuracy whereas 98.31% overall accuracy was attained for the time domain feature, SV_{total}. All features achieved 100% sensitivity from this dataset. In terms of specificity, the time domain feature, SV_{total}, attained up to 97.14% whereas the two LWT features attained 100%. Results suggest that the proposed LWT and SVM model based on the findings in this chapter can serve as a guideline for implementation in actual wearable sensor devices for human fall detection in real time.

CHAPTER IV

FALL DETECTION COMPARISON BETWEEN LIFTING AND CONTINUOUS WAVELET TRANSFORM WITH SUPPORT VECTOR MACHINE

In the previous chapter, the performance of the proposed LWT and SVM model for fall detection was investigated. Results show that the extracted frequency domain features have significant influence on the performance of fall detection. In this chapter, the performance of the proposed LWT combined with the SVM model is further evaluated and compared with other frequency domain features. In particular, the proposed scheme is compared with an existing frequency domain feature extraction method for fall detection, called the continuous wavelet transform (CWT). The performance is evaluated in terms of accuracy, specificity, sensitivity and time computational complexity.

4.1 Introduction

With the development of economics, there is an increasing requirement for healthcare, especially for senior citizens. In 2015, 10% of the population in Thailand were 65 or older and the proportion of old people is still increasing rapidly (World Bank, 2017).

Falls are life threatening risks for old people not only because fall-related injuries, but also because the long-lie posture associated sequelae after a fall. Since the elderly maybe unconscious or unable to call for help (Ozcan, Donat, Gelecek, Ozdirenc, & Karadibak, 2005), it is necessary to develop a fall detection system to help the elderly people avoid falling down and long-lie posture after fall.

Wearable sensor systems make it possible to monitor human movement in daily life without invasion of privacy which is often a concern in camera based fall detection systems (Solanas et al., 2014), (Mazurek, Wagner, & Morawski, 2018). Moreover, wearable sensor is cheap and light. Wearable sensors are often used in the form of sensor fusion that contains accelerometers, gyroscope, pressure sensor and so on.

When it comes to the fall detection algorithms that identify falls from signals obtained from wearable sensors, machine learning is a promising technique. Decision tree (DT) is a basic machine learning method for fall detection. The simplest DT follows the divide-and-conquer strategy (Mingers, 1989). Though DT is a simple machine learning algorithm, it is difficult for DT to deal with continuous segments according to the characteristics of tree model. In addition, the “error accumulation” phenomenon (Q. R. Wang & Suen, 1984) is an inherent drawback of tree model or algorithm. The artificial neural network (ANN) is also a commonly used machine learning method inspired by brain neural network (Xu et al., 2013). ANN can handle big and complex non-linear data, simultaneously. However, this ability may also lead to long-time training process or even the failure of learning.

SVM is another supervised machine learning method. The main idea of SVM is to map nonlinear separable samples into a high dimensional feature space where samples can be divided by a plane called “hyperplane”. Many works applied SVM in fall detection (Pierleoni, Belli, et al., 2015) (Özdemir & Barshan, 2014) due to its generalization ability. However, kernel functions must be chosen according to the specific problem, and there is no standard approach to find the best kernel type. Despite these disadvantages, SVM is still a standard tool which means that there is a mature toolbox for use, for instance, LIBSVM (Chang & Lin, 2011) and LIBLINEAR (Fan, Chang, Hsieh, Wang, & Lin, 2008).

As dimension of the dataset increases in SVM, the data becomes sparser. Furthermore, the large volume of data collected over a long time typically contains only a small fraction needed to identify falls. Hence, feature extraction is essential to reduce the amount of data and computational complexity. (Hossain, Islam, & Ali, 2017) and (Tang & Sazonov, 2014) concentrated on extracting time domain features such as average, maximum, minimum, variance of signals. Time domain features are simple and can achieve a relatively satisfying results. (Hossain et al., 2017) used the mean and standard deviation and reached an accuracy of 96.45%. (Tang & Sazonov, 2014) proposed a time domain data rejection SVM for human postures and activity recognition.

However, signals contain more information than just time domain features. As far as our knowledge, frequency domain features in fall detection were rarely researched. CWT is a classic wavelet transform algorithm to extract time-frequency information of signals. It is an improvement of STFT in terms of multiresolution. Similarly, when the original signals are discrete sequences, DWT can be applied on the sequences. In terms of computational complexity, that of wavelet transform is lighter compared with STFT. Assuming that there is a signal with N samples, the computational complexity of STFT is $O(Nlog_2N)$, and that of CWT is $O(N)$. LWT reduced 50% computational complexity of CWT with similar overall accuracy (Yazar et al., 2013).

Therefore, the main contribution of this chapter include, i) we proposed a fall detection scheme based on a low computational frequency domain method based on LWT combined with SVM using a dataset from waist-mounted accelerometer sensor and ii) we compared it with an existing fall detection method based on CWT using Haar wavelet and a customized wavelet based on average falls. The chapter is organized as follows. Section 4.2 gives a background on the underlying concept of wavelet transform for frequency time domain feature extraction. Then fall detection method based on support vector machine (SVM) is presented. In section 4.3, the dataset and pre-processing method are described. Section 4.4 presents the experiment, results and the discussion of the experiment. The conclusion is given in the last section.

4.2 Method

4.2.1 Continuous Wavelet Transform

Continuous wavelet transform (CWT) is a kind of WT (Daubechies, 1992). It is defined as follows:

$$\mathrm{C}(a,b) = \frac{1}{\sqrt{a}} \int_{+\infty}^{-\infty} f(t)\Psi^{*}\left(\frac{t-b}{a}\right) dt \,, \tag{4. 1}$$

where * denotes the complex conjugate, and *f(t)* represents the function being transformed, the function $\Psi(t)$ is the transforming function which is also called "mother wavelet". The mother wavelet has two important properties. Firstly, $\Psi(t)$ should be compactly supported which implies that $\Psi(t)$ is a finite length function (window). Secondly, mother wavelet can be scaled and shifted by parameters "*a*" and "*b*", respectively. Scaling refers to "stretching" or "compressing" the mother wavelet (Polikar, 1996). Shifting or translation parameter "*b*" means "delay" or "advancing" the wavelet centered by value "*b*" in time-axis.

4.2.2 Lifting-based discrete wavelet transform

As described in section 2.1, lifting wavelet transform (LWT) is an alternative to DWT and an in-place algorithm. Instead of using filters to split, LWT uses the "lazy wavelet" which simply splits the sequence into even and odd. It is the first step for LWT. Secondly, the odd sequence is predicted based on the even sequence. Thirdly, the difference between predicted odd and real odd which is called detail information is used to update the even samples.

4.2.3 Support vector machine

Support Vector Machine (SVM) is a popular machine learning method which was first proposed by Cortes and Vapnik in 1995 (Cortes & Vapnik, 1995). SVM is a supervised machine learning technique because the "label" of instance when training the model is required. Basic SVM is a linear classification model defined in the feature space. However, when combined with a Kernel function, SVM can perform non-linear classification.

4.3 Proposed fall detection method

The proposed fall detection method is shown in
Figure **4.1**. The WT sub-process represents CWT with Haar wavelet and the customized average fall wavelet as well as LWT with Haar wavelet. The sub-process

SVM, for the testing set, is used to classify the testing dataset based on the trained SVM model.

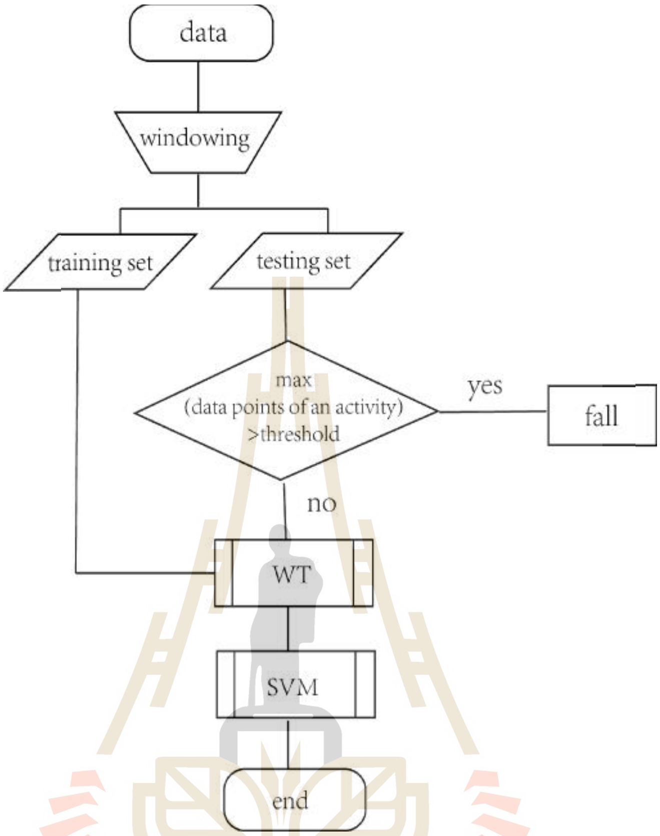


Figure 4.1 Experiment flow chart

The dataset used is collected by a tri-accelerometer attached on subjects' waist. The subjects include 12 volunteers, 5 males and 7 females. Every volunteer applies 13 types of falls and 12 types of ADLs successively with free break intervals. Table 4.1 summarizes all the activity types collected.

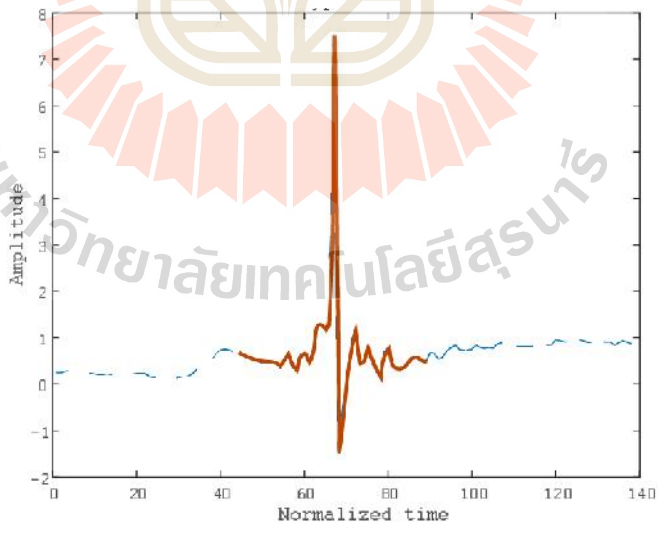
Table 4.1 The type of activities collected

No.	**Falls**	**ADLs**
1	Forward collapse (on knees)	Sitting down on chair
2	Forward collapse (lying down)	Standing up from chair
3	Forward fall (trying to get up for 30s)	Collapsing into a chair
4	Backward collapse (sitting)	Resting against a wall, then sliding vertically down to the end of the sitting position
5	Backward collapse (lying down)	Lying down on a bed
6	Backward collapse (trying to get up for 30s)	Getting up from a bed
7	Sideways collapse (Right)	Jumping vertically
8	Sideways collapse (Left)	Pick up something from the floor
9	Fall from chair (slide)	Bend forward and tie shoe laces
10	Forward fall with recovery (then walking)	Take the lift down
11	Forward fall with recovery (then standing)	Take a lift up
12	Collapsing into a bed	-
13	Fall from bed (try to get up then fall)	-

4.3.1 Performance metrics

In order to evaluate the impact of different inputs on the performance of SVM for fall detection based on tri-accelerometer data, we measured the true positives (TPs) and true negatives (TNs), which correspond to the correctly identified falls (positives) and ADLs (negatives). We also measured false positives (FPs) and false negatives (FNs), which correspond to the false identification of falls and ADLs, respectively. These measurements are for the sake of the following metrics required to evaluate the fall detection metrics: sensitivity (SE), specificity (SP) and accuracy (AC). The definition of these metrics can be found in chapter III.

The results in the previous chapter showed that "Haar" wavelet outperformed Bior2.2 wavelet for fall detection based on single tri-accelerometer sensor. Thus, LWT and CWT in this chapter are based on the Haar wavelet. Furthermore, as did in (Palmerini et al., 2015), we also compared CWT using Haar and a customized mother wavelet based on averaged fall signals.



Amplitude
Normalized time

The acquired data is then compared with a threshold before wavelet transform. Activities whose maximum SV_{total} is higher than the threshold are considered as falls immediately. However, activities with a maximum SV_{total} lower than the threshold need further analysis by the proposed WT-SVM algorithm. By adding a threshold, we have less data which undergo wavelet transform and SVM, and thereby further reducing computation.

It is necessary to have a proper threshold value. There are 296 activities in total, of which half of them are used as training set and the other half are used as testing set. In the training set, 78 of them are falls and the other 70 are ADLs. The average maximum SV_{total} of training falls is used as the threshold. Let the window size denoted by n. Assume that, after windowing, the dataset is { $\mathbf{A_i}$, $\mathbf{F_j}$ | i=1,2,…140, j=1,2,…,156} where $\mathbf{A_i}=\{x_i^1, x_i^2, \ldots, x_i^n\}$ denotes the i^{th} ADL, vector $\mathbf{F_j}=\{y_j^1, y_j^2, \ldots, y_j^n\}$ denotes the j^{th} fall, x_i^l and y_j^l are the l^{th} value of SV_{total} of the ADL and fall dataset, respectively. The training dataset is {$\mathbf{A_1}$, $\mathbf{A_2}$,…, $\mathbf{A_{70}}$, $\mathbf{F_1}$, $\mathbf{F_2}$, …, $\mathbf{F_{78}}$}, and the testing dataset is { $\mathbf{A_{71}}$, $\mathbf{A_{72}}$,… , $\mathbf{A_{140}}$, $\mathbf{F_{79}}$, $\mathbf{F_{80}}$, …, $\mathbf{F_{156}}$}. The threshold is given by

$$Threshold = \frac{\sum_{j=1}^{78} \max(F_j)}{78}, \tag{4. 2}$$

where $max(F_j)$ is the maximum value function in a fall. If the series of $\{s_k^l \mid l = 1, 2, \ldots n\}$, where $s_k^l = \frac{\sum_{j=1}^{78} y_j^l}{78}$ is a series of averaged sample points overall activities. This average series will be used in MATLAB to create a customized fall mother wavelet for CWT.

(3) SVM sub-process after LWT and CWT

In the previous chapter, we found that Haar wavelet performed better than Bior2.2 for fall detection based on tri-accelerometer data (Liang & Usaha, 2017).

Therefore, Haar is used as mother wavelet for CWT and LWT in this chapter. Additionally, the custom average fall wavelet is applied to construct CWT as in (Palmerini et al., 2015) for comparison. The detailed process of SVM after CWT and LWT is shown in

Figure **4.3**.

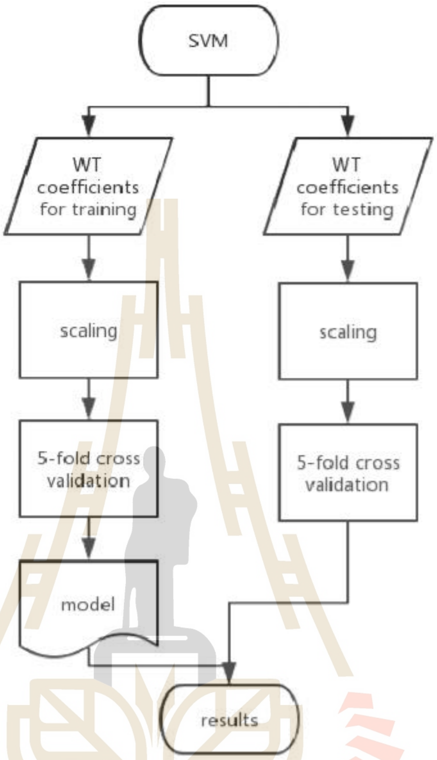


Figure 4.3 SVM detailed flow chart

As shown in

Figure **4.3**, after wavelet transform, the coefficients are the input of SVM. Scaling is for normalization purposes. The next step is a 5-fold cross validation. Cross validation is important for SVM since it prevents overfitting. This experiment uses 5-fold cross validation as suggested in (Chang & Lin, 2011). For the training set, SVM creates a set of parameters which are called model. This model is used to classify the testing data as a fall or an ADL.

4.4 Results and discussion

This experiment uses sensitivity, specificity and accuracy to measure the performance of input. The result shows that LWT cD coefficients with threshold works better than WT-SVM alone (without threshold). The best performance reaches up to 100% of accuracy, sensitivity and specificity which means that every activity in the testing dataset is correctly identified. The worst performance appears in the maximum coefficient of 2-second-window-CWT without threshold. A possible reason may be that most falls occur in a short period of time. Hence, there may not always be enough data within a 2-second window.

4.4.1 CWT with Haar

The objective of this part is to compare the performance of CWT coefficient features. The CWT coefficients are calculated from SV_{total}. Two different features have been derived from the CWT coefficients. For the first feature, all CWT coefficients are chosen as a high dimension feature to represent the activity. For the second feature, only the maximum value of the CWT coefficients is selected as a feature to represent an activity. In particular, let a fall dataset be given by $\mathbf{F_j}=\{y_j^1, y_j^2,\ldots, y_j^n\}$. After CWT, the coefficients become $\mathbf{C_j}=\{c_j^1, c_j^2,\ldots, c_j^n\}$. Thus, these two features are defined as follows: feature-1= $\mathbf{C_j}$,and feature-2 = $max(\mathbf{C_j})$. Table 4.2 shows comparable performance between the two features for CWT (Haar wavelet). More significant improvement can be observed when the threshold for preliminary screening of falls, prior to CWT is employed. The 1- and 2-second windows do not show significant differences.

Table 4.2 The performance of CWT (Haar wavelet) coefficients input into SVM

CWT (Haar) coefficients input into SVM		SVM alone			SVM + Threshold		
		AC (%)	SP (%)	SE (%)	AC (%)	SP (%)	SE (%)
1-second Window	All CWT Coefficients Input Into SVM	59.5	58.6	60.3	90.5	97.1	84.6
	Max. Coefficient Input into SVM	58.1	50.0	65.4	89.2	92.9	85.9
2-second Window	All CWT Coefficients Input Into SVM	55.4	34.3	74.4	90.5	100	82.1
	Max. Coefficient Input Into SVM	56.8	67.1	47.4	92.6	97.1	88.5

4.4.2 CWT customized wavelet and entire and max coefficients

Table 4.3 also compares the performance of CWT coefficients together with SVM (CWT-SVM) as well. However, in this part, the Haar wavelet was replaced by a customized average fall wavelet calculated from described in section 4.3.2.

Table 4.3 The performance of CWT (Customize Wavelet) coefficients input into SVM

CWT (Customized Wavelet) coefficients input into SVM		SVM alone			SVM + Threshold		
		AC (%)	SP (%)	SE (%)	AC (%)	SP (%)	SE (%)
1-second Window	All CWT Coefficients Input Into SVM	59.5	58.6	60.3	89.9	91.4	88.5
	Max. Coefficient Input Into SVM	64.2	72.9	56.4	91.9	95.7	88.5
2-second Window	All CWT Coefficients Input Into SVM	54.7	44.3	64.1	89.9	97.1	83.3
	Max. Coefficient Input Into SVM	50.7	18.6	79.5	90.5	100	82.1

Similar to Table 4.2, the threshold prior to CWT performs significantly better than the method without threshold. In Table 4.3, the best total accuracy is 91.9% using the 1-second window with threshold and max-coefficient CWT features.

4.4.3 LWT (Haar wavelet) coefficients

Table 4.4 shows the performance comparison between using LWT with detailed coefficients (cD) and with both approximate and detailed coefficients (cA and cD respectively) as features.

Table 4.4 The performance of LWT (Haar) coefficients input into SVM

LWT (Haar) coefficients input into SVM		**SVM alone**			**SVM + Threshold**		
		AC (%)	SP (%)	SE (%)	AC (%)	SP (%)	SE (%)
1-second Window	LWT coefficients cD input into SVM	60.1	72.6	48.7	100	100	100
	LWT coefficients cD and cA Input Into SVM	62.8	58.6	66.7	90.5	100	82.1
2-second Window	LWT coefficients cD input into SVM	59.5	55.7	62.8	90.5	100	82.1
	LWT coefficients cD and cA input into SVM	58.1	47.1	67.9	89.2	94.3	84.6

The 1-second-window LWT cD coefficients with threshold outperform the other features. In this situation, the accuracy, specificity and sensitivity achieved 100%. In other words, using LWT (Haar) coefficients cD as features, SVM is able to classify all the testing activities correctly. Interestingly, the 1-second window with LWT cD and cA coefficients has the same number of input data as the 2-second window with LWT cD coefficients, and they have exactly the same results. Moreover, cD consistently outperforms cD and cA in same scenario. It may imply that cD and cA feature is redundant compared with cD.

4.4.4 Computational complexity of LWT and CWT

Table 4.5 The time consumption of LWT and CWT

Time	**Time/loop(sec)**	**Create Mother wavelet (sec)**	**Total(sec)**
LWT(Haar)	6.9	-	6.9
CWT(Haar)	10.8	-	10.8
CWT(Custom Wavelet)	10.8	2.8	13.6

Table 4.5 illustrates the execution time for each algorithm using MATLAB

R2014b @Windows 10.1. Because it is difficult to maintain a controlled performance throughout the duration of simulation on the computer, we measured the one loop (i.e. a 1-second window) of LWT and CWT, respectively. LWT shows shorter computational time than CWT using Haar and customized fall wavelets.

4.5 Summary

In this chapter, we developed a fall detection algorithm based on the proposed LWT combined with the SVM model and compared it with additional frequency domain features based on CWT-SVM with a new dataset from a tri-accelerometer sensor. Since the 2-second window did not consistently contain sufficient amount of data when the position of the peak is located towards the end of the dataset, the 1-second window performs better than the 2-second window. This is the case for both LWT-SVM and CWT-SVM. Given an average of the maximum SV_{total} of training set falls as a threshold prior to the wavelet transform, the WT-SVM algorithm shows a significant improvement in accuracy from around 60% to over 90%.

As for the results of CWT-SVM, we notice that the total accuracy is around 90% with threshold, and under 60% for most CWT without threshold cases. The specificity is typically higher than sensitivity, implying that CWT coefficients features tend to classify testing activities as falls. Though not as good as LWT in the same condition, in 2-second window with threshold scenario, CWT has better result than LWT.

LWT cD coefficients with Haar using the 1-second-window with threshold can achieve the highest accuracy, specificity and sensitivity of 100%. However, the best performance of CWT is 92.6% accuracy, 97.1% sensitivity and 88.5% specificity, is attained from the CWT with maximum-CD coefficient with Haar using the 2-second-window with threshold. Therefore, despite its light computational requirement, LWT can outperform CWT frequency domain features. LWT also provides the shortest computational time per window when compared with CWT using Haar and customized fall wavelets.

CHAPTER V

CONCLUSION AND FUTURE WORK

5.1 Conclusion

This thesis proposed lightweight algorithm to extract frequency domain features for fall detection. The contribution of this work mainly lays on the light computational cost frequency domain feature extraction method we proposed for fall detection. Wavelet transforms were used in experiments to extract time-frequency domain features. In chapter III, the proposed time-frequency domain features extracted by LWT were compared with the time domain root sum square (RSS) features. Various SVM models were investigated to determine the best possible model to be combined with the LWT feature extraction. The best performance was achieved by the level 1 detailed coefficients (cD1) LWT with Haar wavelets using an 8% threshold, which achieved a total accuracy, sensitivity and specificity of 100%.

Later in chapter IV, additional frequency domain features, namely, the CWT coefficients were compared with LWT coefficients combined with SVM. In particular, the CWT based on Haar wavelet and CWT based on customized average fall wavelet were compared with the proposed LWT with Haar wavelet. The proposed scheme outperformed the CWT schemes for extracting fall detection frequency domain features. The best performance was attained from the LWT scheme using Haar wavelet with level 1 coefficients (cD1) using a 1-second window with a threshold, achieving the highest accuracy, specificity and sensitivity of 100%.

LWT also showed a significant advantage over CWT in respect of time computational complexity. As a feature extraction technique for SVM based fall detection, LWT was almost twice as fast as CWT in the same scenario.

Despite the advantages of the proposed LWT-SVM method for fall detection, there are certain limitations.

(1) Wearable sensors will encounter the dilemma of battery power usage despite its low computational requirement onboard in order to achieve long battery operation in wearable sensors. The effect of the proposed LWT-SVM on the battery lifetime is not yet investigated.

(2) The dataset used for experiments in this thesis were simulated falls from young and healthy volunteers. Thus, the results of this work may not fully represent the realistic falls of the elderly in their daily lives.

(3) The algorithm was designed based on data collected by an accelerometer sensor alone for off-line fall detection. In order to achieve full online fall detection, additional sensory data from other types of sensors as well as sensor fusion may be needed.

5.2 Future work

In the future, the issues worthwhile to investigate are the followings.

5.2.1 Developing the related hardware

In this thesis, two datasets were used to evaluate the proposed algorithm. However, the dataset may be collected from actual hardware implementation which has not been considered in this thesis. Issues related to the effects of the proposed algorithm on the battery lifetime of the wearable sensor is also a significant matter for investigation.

5.2.2 Multi-sensor nodes may works better in some cases

The number of sensor nodes is also worthwhile investigating. In (Özdemir, 2016), the influence of the number of sensors on the performance of SVM was studied. Whether the number of sensors have the same influence on our proposed algorithm and the types of sensors required for the best performance should be investigated.

5.2.3 Real world fall dataset

The experiments in this thesis are entirely based on simulated and controlled fall datasets which have been collected from young and healthy volunteers in the laboratory. Thus, even though the proposed method performs well in this thesis, it remains uncertain whether or not this is the case in presence of real falls from the elderly. Due to the lack of real world falls and activities in daily living environment, the dataset of such nature will be vital for validating any fall detection scheme.

REFERENCES

Abbate, S., Avvenuti, M., Bonatesta, F., Cola, G., Corsini, P., & Vecchio, A. (2012). A smartphone-based fall detection system. **Pervasive and Mobile Computing**, 8(6), 883-899.

Aziz, O., Musngi, M., Park, E. J., Mori, G., & Robinovitch, S. N. (2017). A comparison of accuracy of fall detection algorithms (threshold-based vs. machine learning) using waist-mounted tri-axial accelerometer signals from a comprehensive set of falls and non-fall trials. **Medical and Biological Engineering and Computing**, 55(1), 45-55.

Baker, J. (1975). The DRAGON system : An overview. **IEEE Transactions on Acoustics, Speech, and Signal Processing**, 23(1), 24-29.

Banaee, H., Ahmed, M. U., & Loutfi, A. (2013). Data mining for wearable sensors in health monitoring systems: a review of recent trends and challenges. **Sensors**. 13(12), 17472-17500.

Bank, W. (2017). Population ages 65 and over (% of total). Retrieved from **World Bank**. Doi: https://data.worldbank.org/indicator/SP.POP.65UP.TO.ZS.

Bilski, P., Mazurek, P., Wagner, J., & Winiecki, W. (2015). Application of decision trees to the fall detection of elderly people using depth based sensors. **XXI IMEKO World Congress" Measurement in Research and Industry"**.15(6), 24-26.

Björklund, S., Petersson, H., & Hendeby, G. (2015). Features for micro-Doppler based activity classification. **IET Radar, Sonar and Navigation.** 9(9), 1181-1187.

Bourke, A., O'brien, J., & Lyons, G. (2007). Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm. **Gait and Posture.** 26(2), 194-199.

Boyd, S., & Vandenberghe, L. (2004). **Convex optimization**. Cambridge university press, UK.

Carlsson, T. (2015). **Individualized Motion Monitoring by Wearable Sensor: Pre-impact fall detection using SVM and sensor fusion**. School of Technology and Health, Sweden.

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: a library for support vector machines. **ACM Transactions on Intelligent Systems and Technology (TIST).** 2(3), 27.

Chui, C. K. (1992). **Wavelets: a tutorial in theory and applications**. Academic Press, USA.

Colkesen, T. K. a. I. (2012). The effects of training set size for performance of support vector machines and decision trees. **International Symposium on Supercritical Fluids**. 10(5), 10-13

Cortes, C., & Vapnik, V. (1995). Support-vector networks. **Machine Learning**, 20(3), 273-297.

Daubechies, I. (1992). **Ten lectures on wavelets**. SIAM, Philadelphia, Pa, USA.

Daubechies, I., & Sweldens, W. (1998). Factoring wavelet transforms into lifting steps. **Journal of Fourier Analysis and Applications**. 4(3), 247-269.

Vetterli, M., (2013). **Subband image coding**. Springer, Germany.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., & Lin, C.-J. (2008).

LIBLINEAR: A library for large linear classification. **Journal of Machine Learning Research**.. 9(8), 1871-1874.

Hanai, Y., Nishimura, J., & Kuroda, T. (2009). Haar-like filtering for human activity recognition using 3D accelerometer. **Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop**. 09(13), 675-678.

He, Z., & Jin, L. (2009). Activity recognition from acceleration data based on discrete consine transform and SVM. **2009 IEEE International Conference on Systems, Man and Cybernetics**. 09 (10), 5041-5044.

Heisele, B., Ho, P., & Poggio, T. (2001). Face recognition with support vector machines: Global versus component-based approach. **Eighth IEEE International Conference on Computer Vision**. 7(8), 688-694.

Hossain, S. F., Islam, M. Z., & Ali, M. L. (2017). Real time direction-sensitive fall detection system using accelerometer and learning classifier. **IEEE International Conference on Advances in Electrical Engineering**. 9(4), 99-104.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2003). **A practical guide to support vector classification**. National Taiwan University, Taipei. 4(13), 1-16

Kianoush, S., Savazzi, S., Vicentini, F., Rampa, V., & Giussani, M. (2015). Leveraging RF signals for human sensing: fall detection and localization in human-machine shared workspaces. **IEEE International Conference on Industrial Informatics**. 2015(13), 1456-1462.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. **International Jiont Conference on Artifical Inteligence**. 14(2), 1137-1145.

Koller, D., & Friedman, N. (2009). **Probabilistic graphical models: principles and techniques**. MIT press.

Krishnan, M., Neophytou, C. P., & Prescott, G. (1994). Wavelet transform speech recognition using vector quantization, dynamic time warping and artificial neural networks. *Center for Excellence in Computer Aided Systems Engineering and Telecommunications and Information Science Laboratory*.

Kwolek, B., & Kepski, M. (2014). Human fall detection on embedded platform using depth maps and wireless accelerometer. **Computer Methods and Programs in Biomedicine**, 117(3), 489-501.

Lara, O. D., & Labrador, M. A. (2013). A survey on human activity recognition using wearable sensors. **IEEE Communications Surveys and Tutorials**, 15(3), 1192-1209.

Li, Z., Wei, Z., Yue, Y., Wang, H., Jia, W., Burke, L. E., Sun, M. (2015). An adaptive hidden markov model for activity recognition based on a wearable multi-sensor device. **Journal of Medical Systems**, 2015(39), 56-66.

Liang, H., & Usaha, W. (2017). Fall detection using lifting wavelet transform and support vector machine. **Federated Conference on Computer Science and Information Systems**. 17(9), 877-882.

Liu, S.-H., & Cheng, W.-C. (2012). Fall detection with the support vector machine during scripted and continuous unscripted activities. **Sensors**, 12(9), 12301-12316.

Ma, C., Randolph, M. A., & Drish, J. (2001). A support vector machines-based rejection technique for speech recognition. **IEEE International Conference on Acoustics, Speech, and Signal Processing**. 2001(1), 381-384.

Mazurek, P., Wagner, J., & Morawski, R. Z. (2018). Use of kinematic and mel-cepstrum-related features for fall detection based on data from infrared depth sensors. **Biomedical Signal Processing and Control**, 40(2018), 102-110.

Mingers, J. (1989). An empirical comparison of selection measures for decision-tree induction. **Machine Learning**, 3(4), 319-342.

Myles, A. J., Feudale, R. N., Liu, Y., Woody, N. A., & Brown, S. D. (2004). An introduction to decision tree modeling. **Journal of Chemometrics**, 18(6), 275-285.

Noshadi, H., Dabiri, F., Ahmadian, S., Amini, N., & Sarrafzadeh, M. (2013). HERMES: mobile system for instability analysis and balance assessment. **ACM Transactions on Embedded Computing Systems (TECS)**, 12(1), 57:1-57:24.

Nukala, B. T., Shibuya, N., Rodriguez, A., Tsay, J., Lopez, J., Nguyen, T., .Lie, D. Y.C. (2014). An efficient and robust fall detection system using wireless gait analysis sensor with artificial neural network (ANN) and support vector machine (SVM) algorithms. **Open J. Appl. Biosens**, 2014(3), 29-39.

Ozcan, A., Donat, H., Gelecek, N., Ozdirenc, M., & Karadibak, D. (2005). The relationship between risk factors for falling and the quality of life in older adults. **BMC Public Health**, 5(1), 90:1-90:6.

Özdemir, A. T. (2016). An analysis on sensor locations of the human body for wearable fall detection devices: Principles and practice. **Sensors**, 16(8), 1161:1-1161:25.

Özdemir, A. T., & Barshan, B. (2014). Detecting falls with wearable sensors using machine learning techniques. **Journal of Sensors**, 14(6), 10691-10708.

Palmerini, L., Bagalà, F., Zanetti, A., Klenk, J., Becker, C., & Cappello, A. (2015). A wavelet-based approach to fall detection. **Journal of Sensors**, 15(5), 11575-11586.

Paradiso, J., Hu, E., & Hsiao, K. (1999). The CyberShoe: a wireless multisensor interface for a dancer's feet. **Proceedings of International Dance and Technology**, 1999(99), 57-60.

Parkka, J., Ermes, M., Korpipaa, P., Mantyjarvi, J., Peltola, J., & Korhonen, I. (2006). Activity classification using realistic data from wearable sensors. **IEEE Transactions on Information Technology in Biomedicine**, 10(1), 119-128.

Pierleoni, P., Belli, A., Palma, L., Pellegrini, M., Pernini, L., & Valenti, S. (2015). A high reliability wearable device for elderly fall detection. **Journal of Sensors**. 15(8), 4544-4553.

Pierleoni, P., Pernini, L., Belli, A., Palma, L., Valenti, S., & Paniccia, M. (2015). SVM-based fall detection method for elderly people using Android low-cost smartphones. **IEEE Conference on Sensors Applications Symposium**, 15(4), 1-5.

Polikar, R. (1996). **The wavelet tutorial**. Academic Press, USA.

Quinlan, J. R. (1986). Induction of decision trees. **Machine Learning**, 1(1), 81-106.

Quinlan, S., & Khatib, O. (1993). Elastic bands: Connecting path planning and control. **IEEE International Conference on Robotics and Automation**. 93(5), 802-807.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. **Proceedings of the IEEE**, 77(2), 257-286.

Scholkopf, B., & Smola, A. J. (2001). **Learning with kernels: support vector machines, regularization, optimization, and beyond**. MIT Press, USA.

Shibuya, N., Nukala, B. T., Rodriguez, A., Tsay, J., Nguyen, T. Q., Zupancic, S., & Lie, D. Y. (2015). A real-time fall detection system using a wearable gait analysis sensor and a support vector machine (svm) classifier. **Eighth International Conference on Mobile Computing and Ubiquitous Networking**, 1(8), 66-67.

Shin, I., Son, J., Ahn, S., Ryu, J., Park, S., Kim, J., .Kim, Y. (2015). A novel short-time fourier transform-based fall detection algorithm using 3-axis accelerations. **Mathematical Problems in Engineering**, 2015(2015), 1-7.

Solanas, A., Patsakis, C., Conti, M., Vlachos, I. S., Ramos, V., Falcone, F., Perrea, D. N. (2014). Smart health: a context-aware health paradigm within smart cities. **IEEE Communications Magazine**, 52(8), 74-81.

Su, B. Y., Ho, K., Rantz, M. J., & Skubic, M. (2015). Doppler radar fall activity detection using the wavelet transform. **IEEE Transactions on Biomedical Engineering**, 62(3), 865-875.

Sweldens, W. (1998). The lifting scheme: A construction of second generation wavelets. **SIAM Journal on Mathematical Analysis**, 29(2), 511-546.

Tang, W., & Sazonov, E. S. (2014). Highly accurate recognition of human postures and activities through classification with rejection. **IEEE Journal of Biomedical and Health Informatics**, 18(1), 309-315.

Tong, L., Song, Q., Ge, Y., & Liu, M. (2013). HMM-based human fall detection and prediction method using tri-axial accelerometer. **IEEE Sensors Journal**, 13(5), 1849-1856.

Ustuner, M., Sanli, F., & Abdikan, S. (2016). Balanced vs imbalanced training data: classifying rapideye data with support vector machines. **International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences**, XLI-B7(2016), 379-384.

Vetterli, M. (1984). Multi-dimensional sub-band coding: Some theory and algorithms. **Signal Processing**, 6(2), 97-112.

Wang, Q. R., & Suen, C. Y. (1984). Analysis and design of a decision tree based on entropy reduction and its application to large character set recognition. **IEEE Transactions on Pattern Analysis and Machine Intelligence**(4), 406-417.

Wang, Z., Jiang, M., Hu, Y., & Li, H. (2012). An incremental learning method based on probabilistic neural networks and adjustable fuzzy clustering for human activity recognition by using wearable sensors. **IEEE Transactions on Information Technology in Biomedicine**, 16(4), 691-699.

Wijaya, A. I., Prihatmanto, A. S., & Wijaya, R. (2016). Shesop Healthcare: Stress and influenza classification using support vector machine kernel. **ArXiv e-prints,** 16(7), 1-6.

Wójtowicz, B., Dobrowolski, A., & Tomczykiewicz, K. (2015). Fall detector using discrete wavelet decomposition and SVM classifier. **Metrology and Measurement Systems**, 22(2), 303-314.

Xu, M., Wong, T. C., & Chin, K. S. (2013). Modeling daily patient arrivals at Emergency Department and quantifying the relative importance of contributing variables using artificial neural network. **Decision Support Systems**, 54(3), 1488-1498.

Yazar, A., Keskin, F., Töreyin, B. U., & Çetin, A. E. (2013). Fall detection using single-tree complex wavelet transform. **Pattern Recognition Letters**, 34(15), 1945-1952.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. **The Bulletin of Mathematical Biophysics**, *5*(4), 115-133.

Pannurat, N., Thiemjarus, S., & Nantajeewarawat, E. (2017). A hybrid temporal reasoning framework for fall monitoring. **IEEE Sensors Journal**, 17(6), 1749-1759.

APPENDIX A

DATA USED IN CHAPTER III

Dataset used in chapter III

There are more examples about the dataset used in chapter III. Only 10 falls and 10 ADLs are displayed below. In addition, only LWT with Haar wavelet was used to illustrate the data processing since LWT with Bior2.2 follows a similar method.

The accelerometer was attached on the waist of the subjects.

A.1 Simulation data

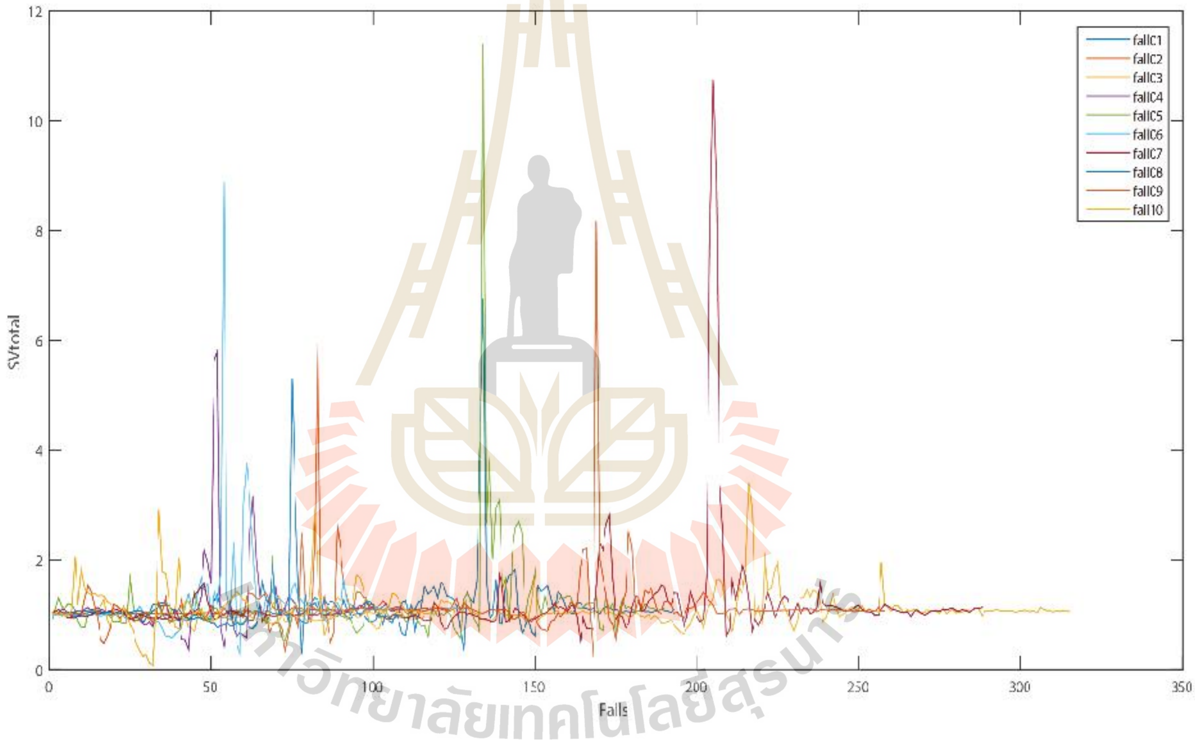


Figure A.1 The simulation falls SV_{total} (fall-01-acc to fall-10-acc)

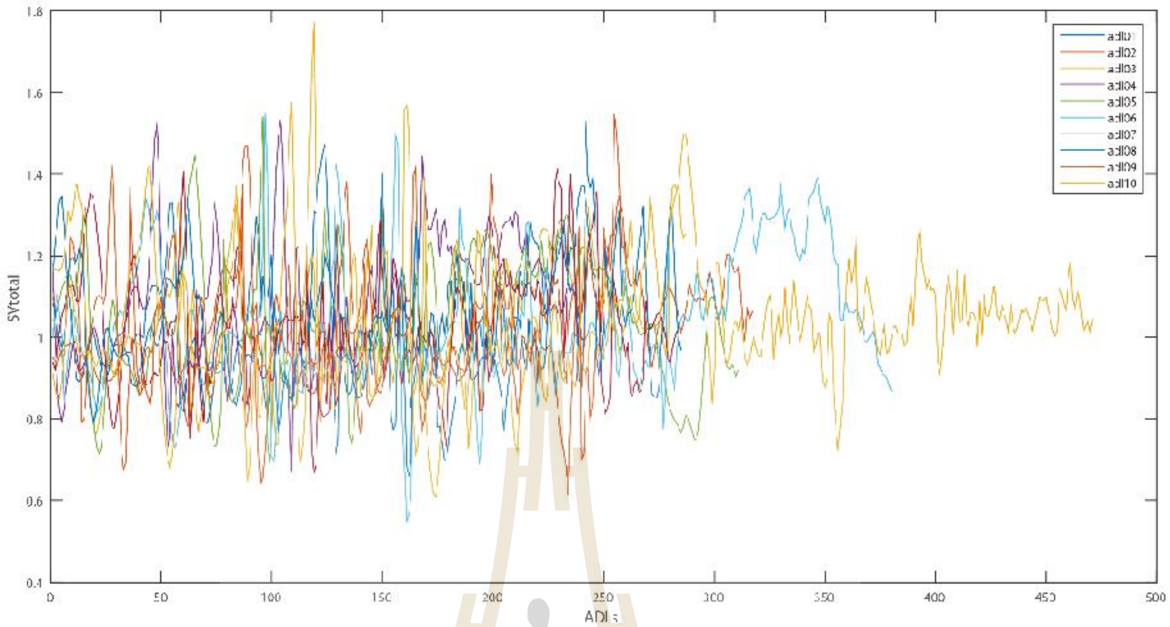


Figure A.2 The simulation ADLs SV_{total} (adls-01-ac to adls-10-acc)

A.2 Simulation data after LWT

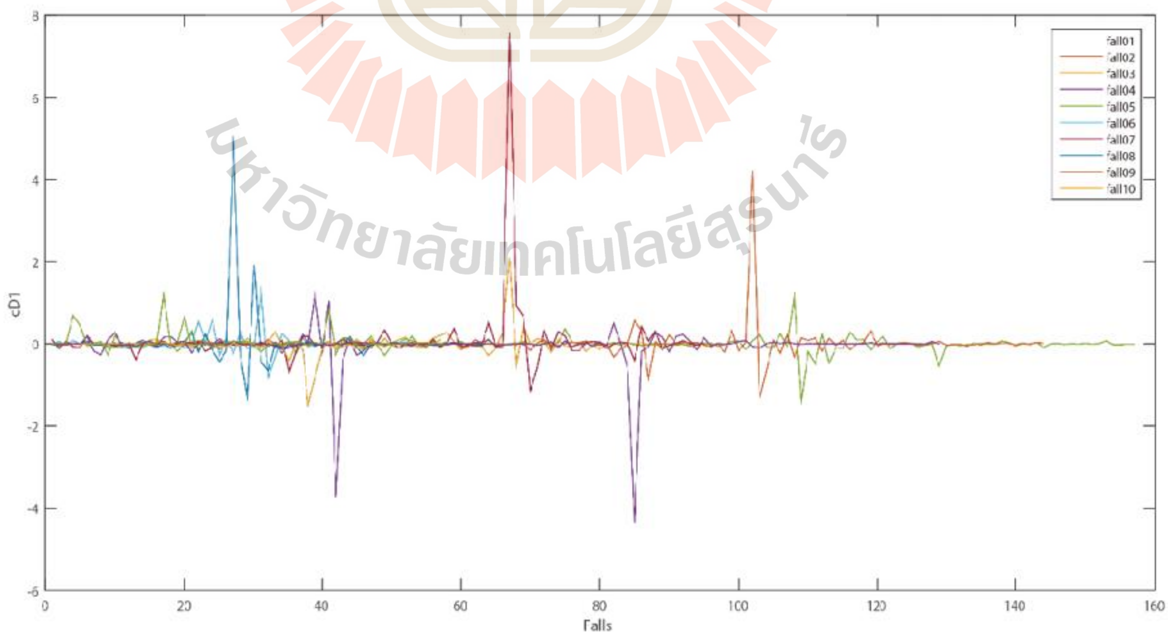


Figure A.3 Falls after 1-level LWT (Haar) cD1

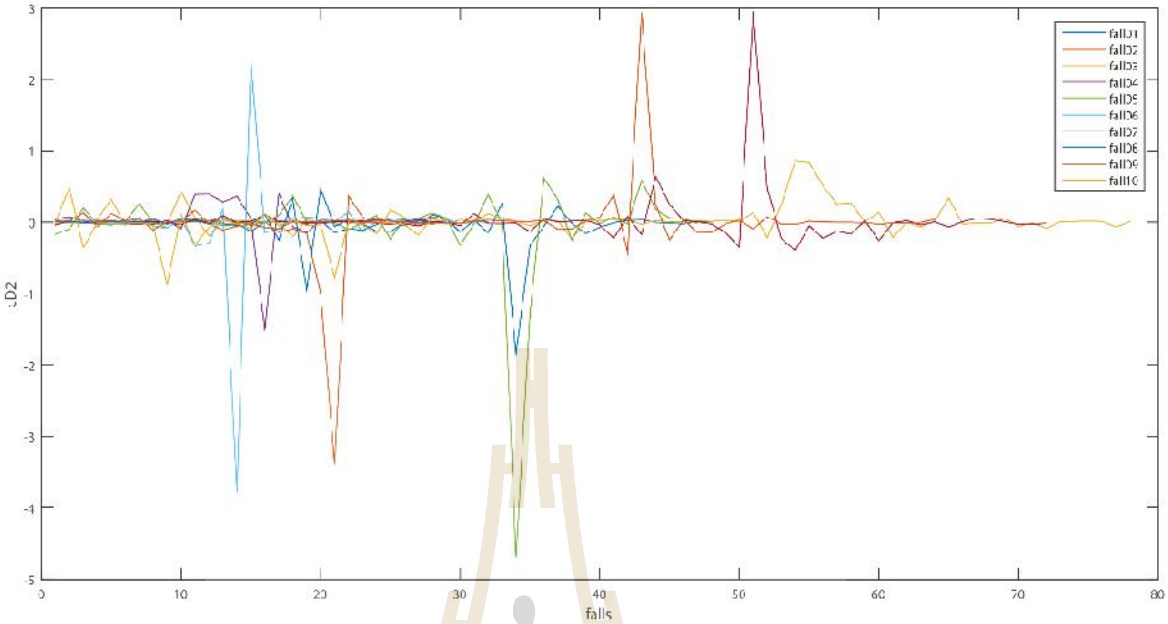


Figure A.4 Falls : SV_{total} after 2-level LWT (Haar) cD2

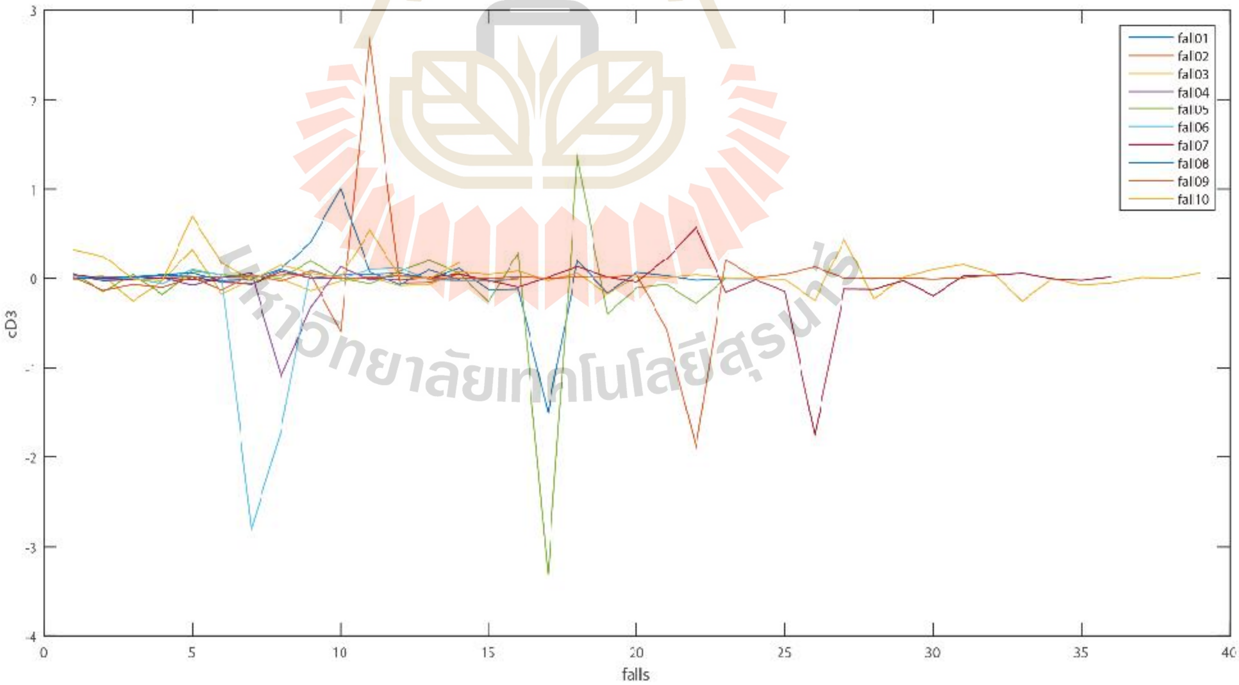


Figure A.5 Falls : SV_{total} after 3-level LWT (Haar) cD3

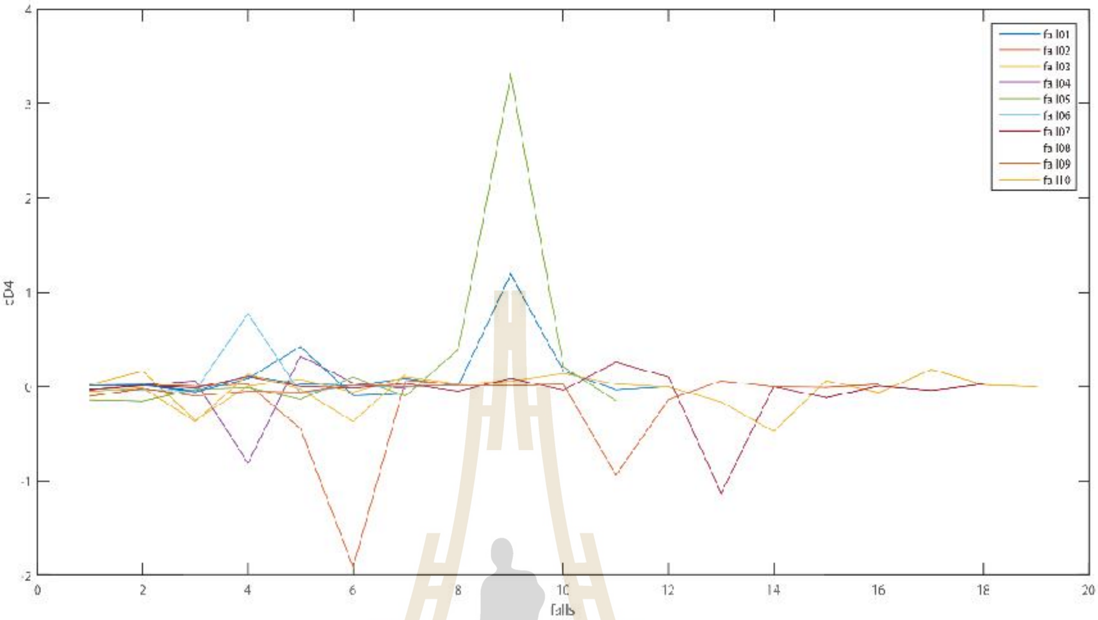


Figure A.6 Falls : SV_{total} after 4-level LWT (Haar) cD4

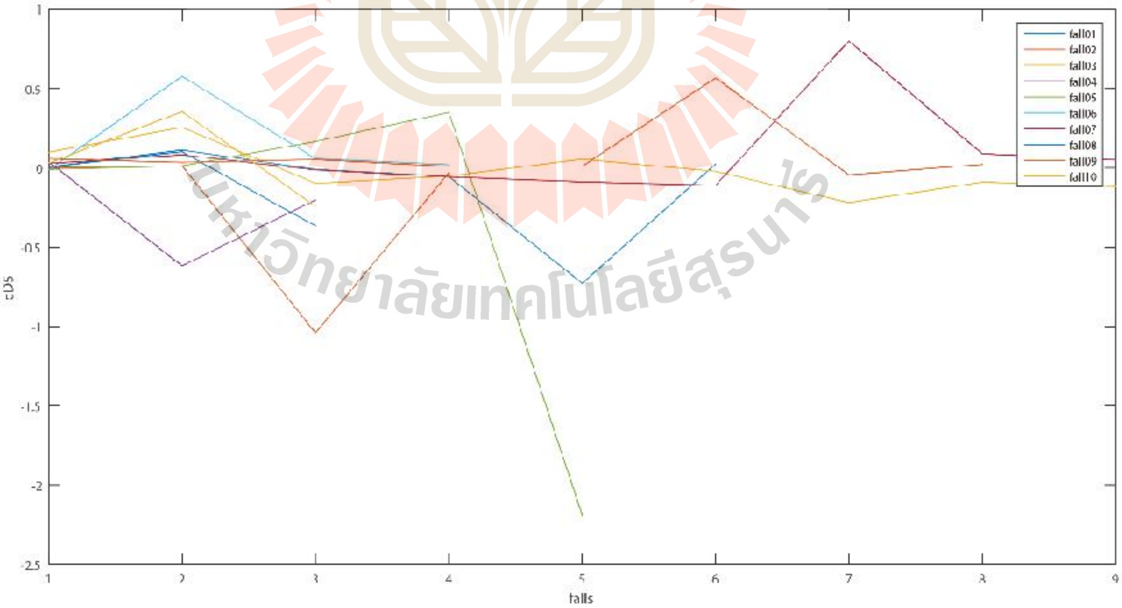


Figure A.7 Falls : SV_{total} after 5-level LWT (Haar) cD5

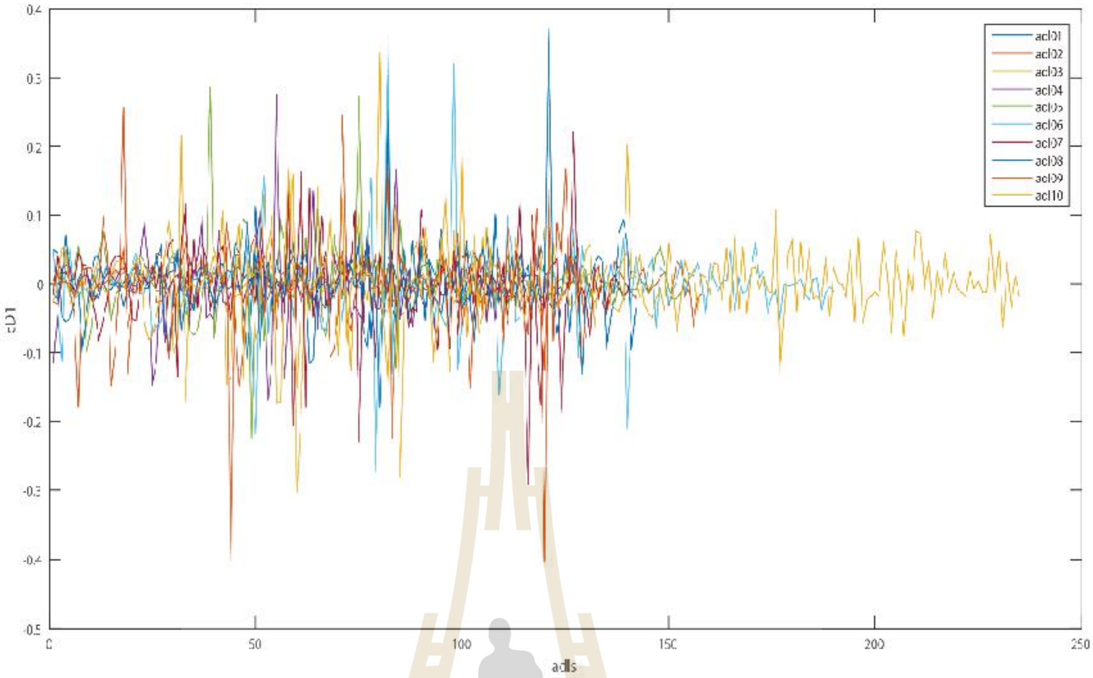


Figure A.8 ADLs after 1-level LWT (Haar) cD1

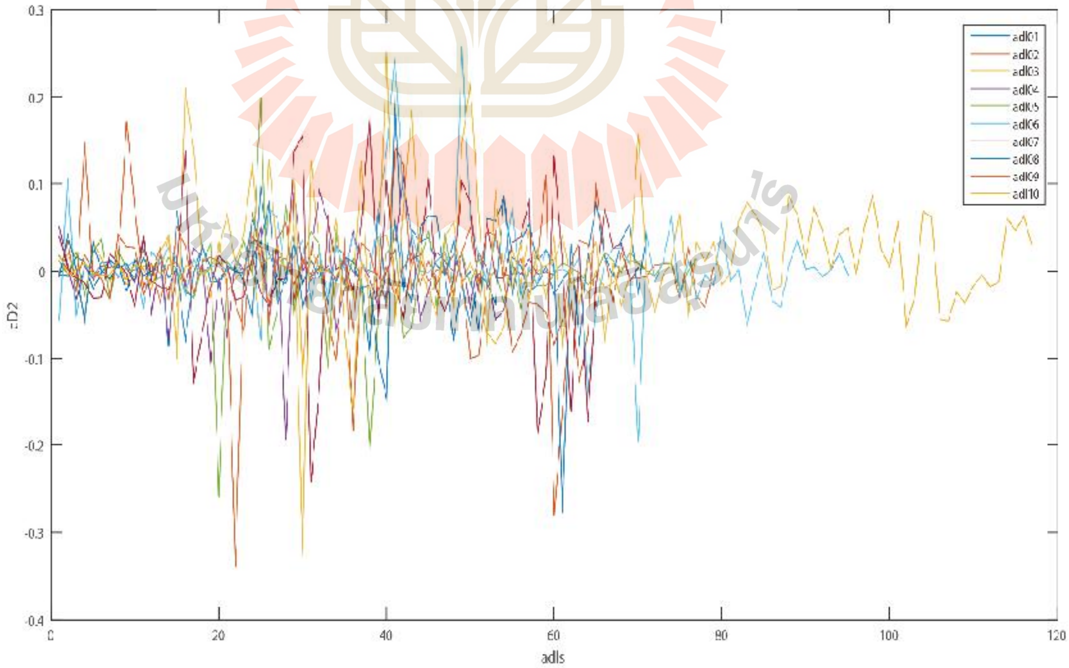


Figure A.9 ADLs : SV_{total} after 2-level LWT (Haar) cD2

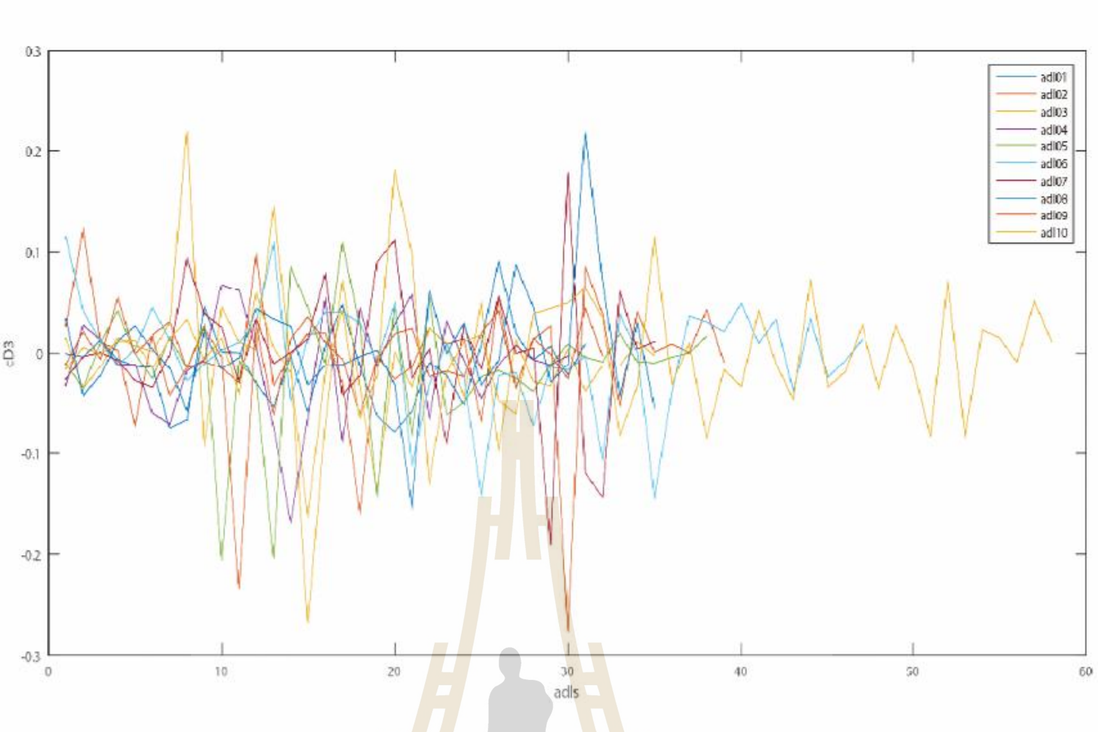


Figure A.10 ADLs : SV_{total} after 3-level LWT (Haar) cD3

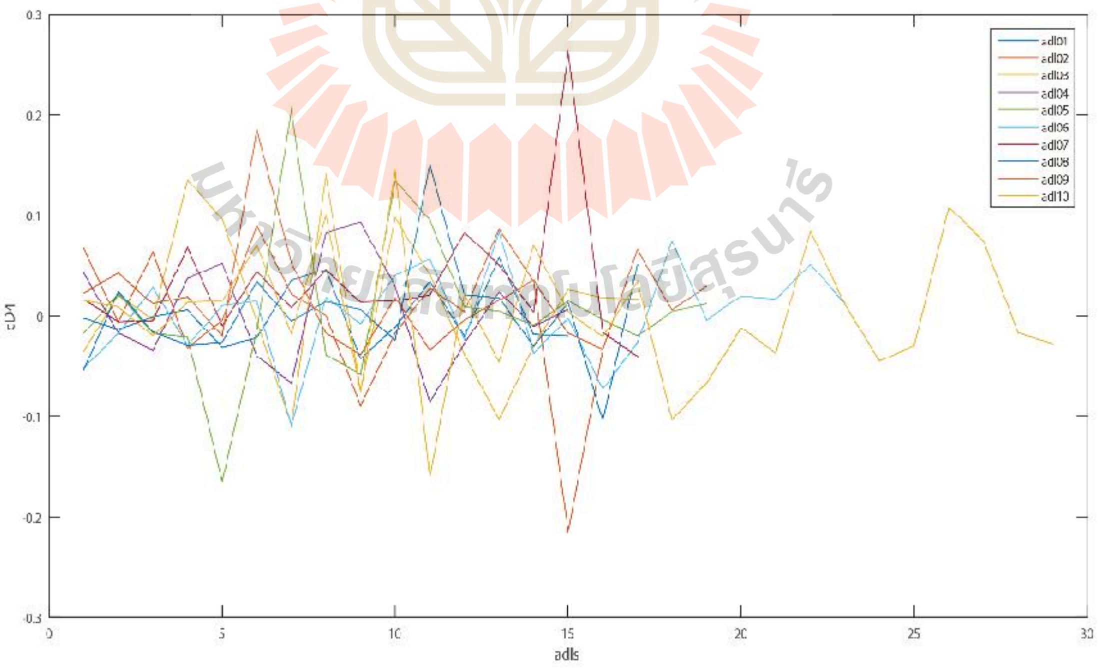


Figure A.11 ADLs : SV_{total} after 4-level LWT (Haar) cD4

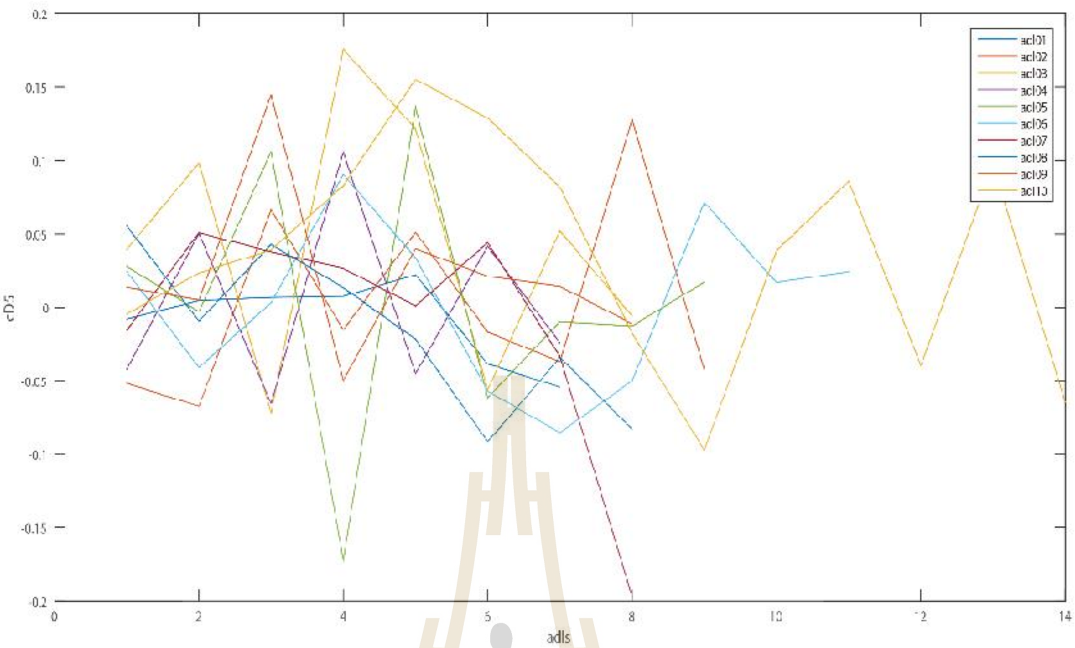


Figure A.12 ADLs : SV_{total} after 5-level LWT (Haar) cD5

APPENDIX B

DATASET USED IN CHAPTER IV

Dataset used in chapter IV

Only the dataset from one subject is illustrated as an example below. Other subjects display similar movements.

B.1 Simulation data SV_{total}

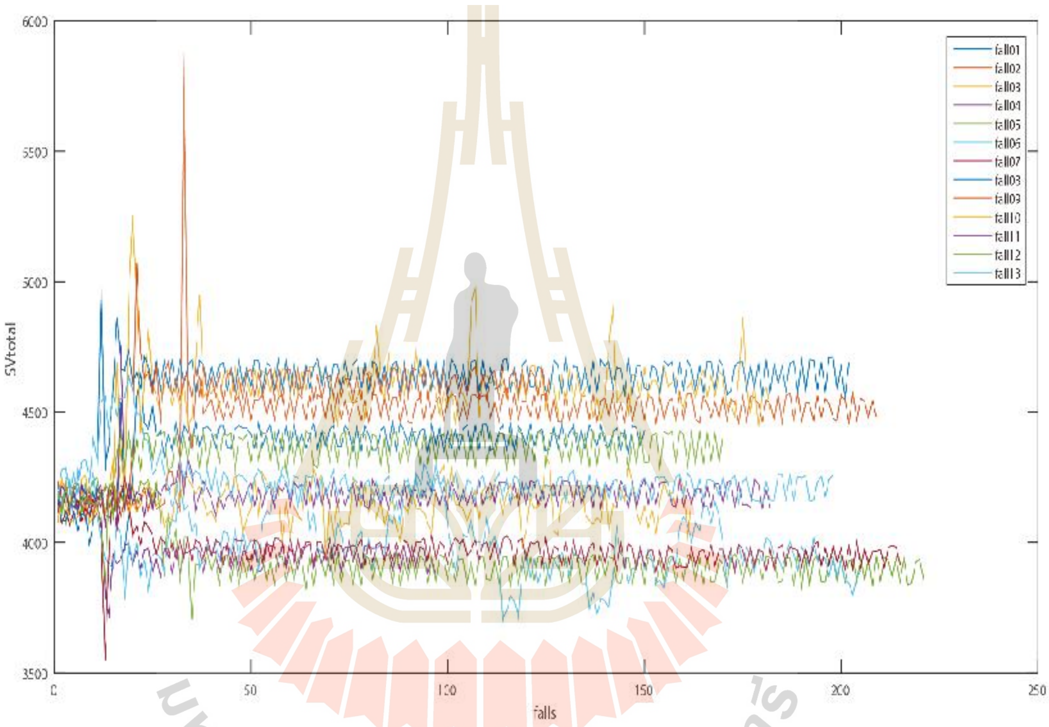


Figure B.1 Falls : SV_{total}

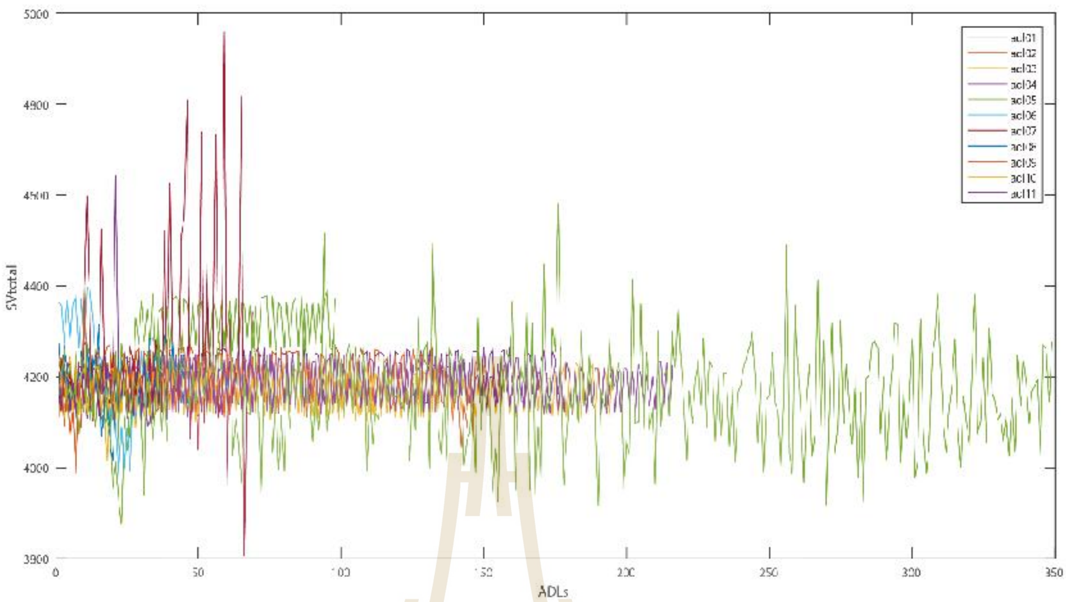


Figure B.2 ADLs : SV_{total}

B.2 Simulation data SV_{total} after a 60 data points window

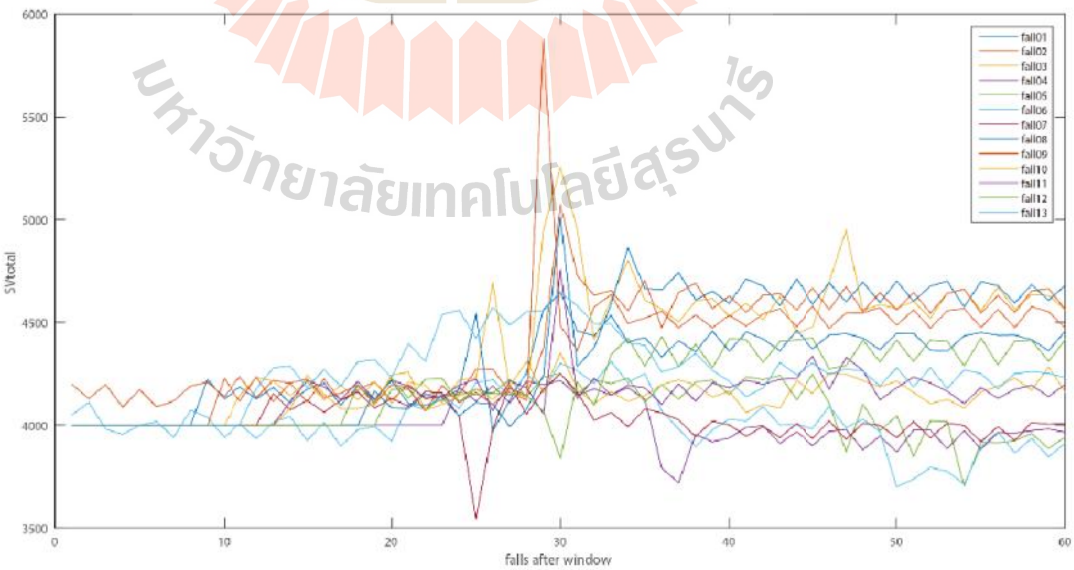


Figure B.3 Falls after a window

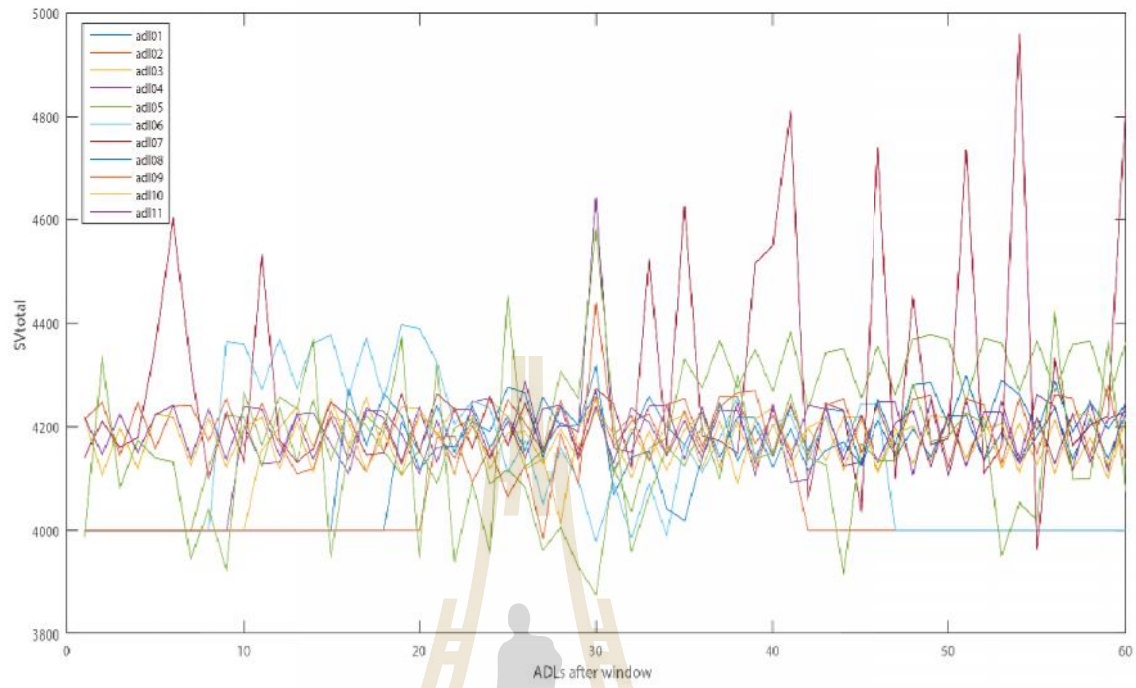


Figure B.4 ADLs after a window

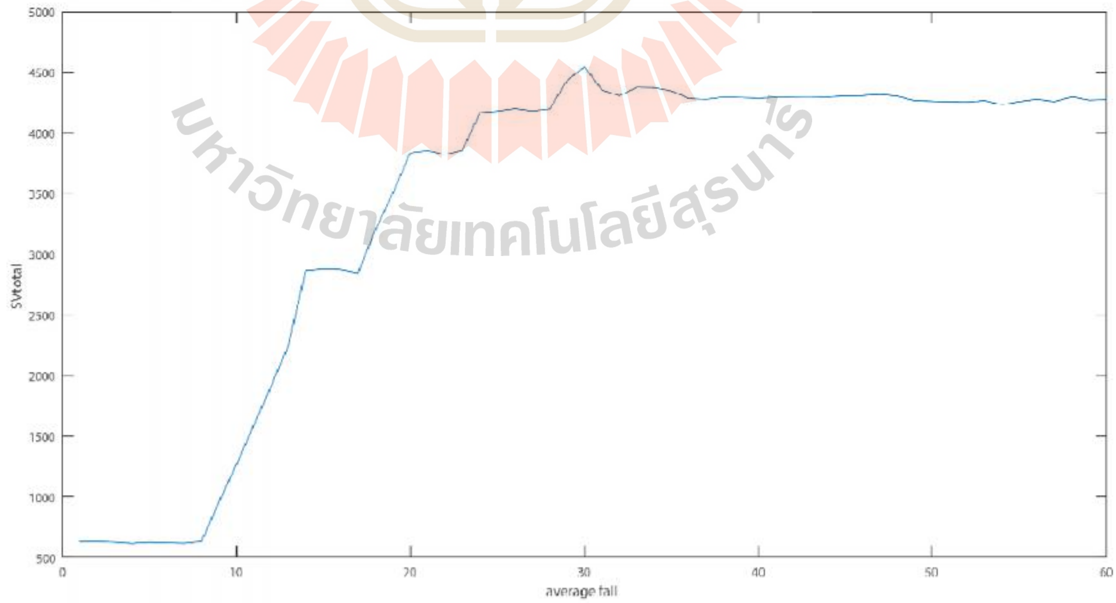


Figure B.5 Average fall after a window

B.3 Simulation data SV_{total} after LWT cD

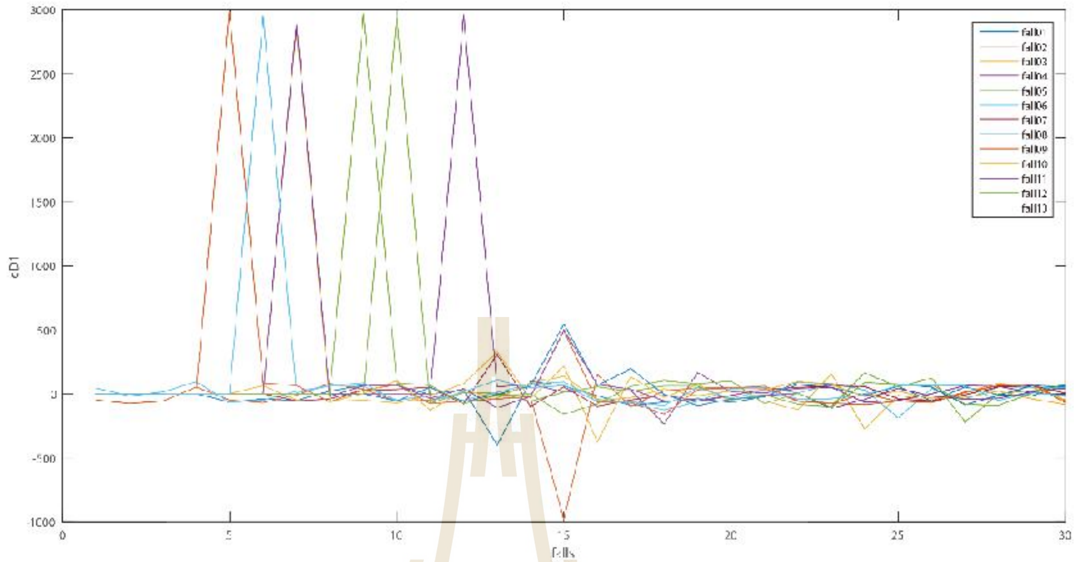


Figure B.6 Falls : SV_{total} after LWT (Haar) cD1

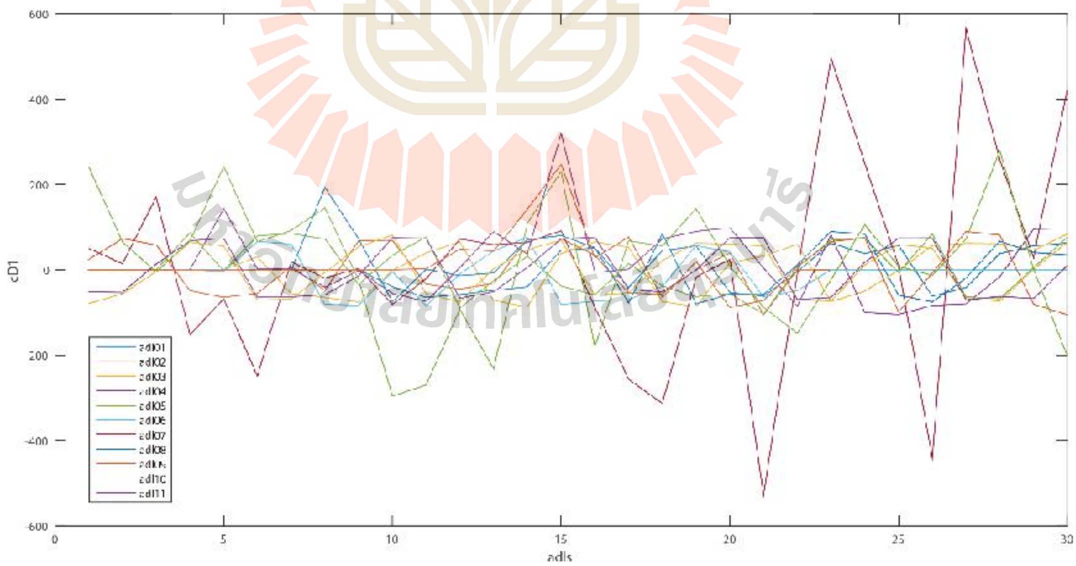


Figure B.7 ADLs : SV_{total} after LWT (Haar) cD1

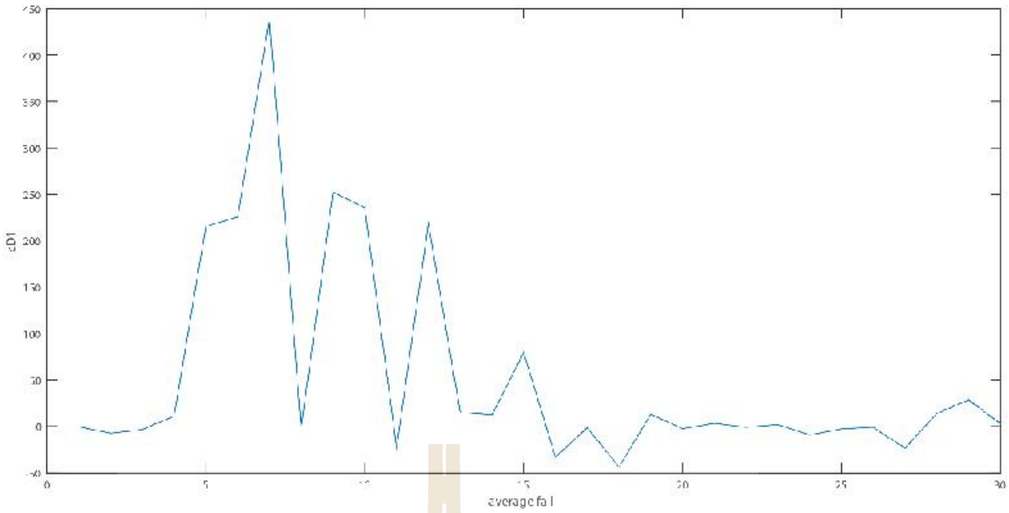


Figure B.8 Average fall SV_{total} after LWT (Haar) cD1

B.4 Simulation data SV_{total} after CWT coefficients

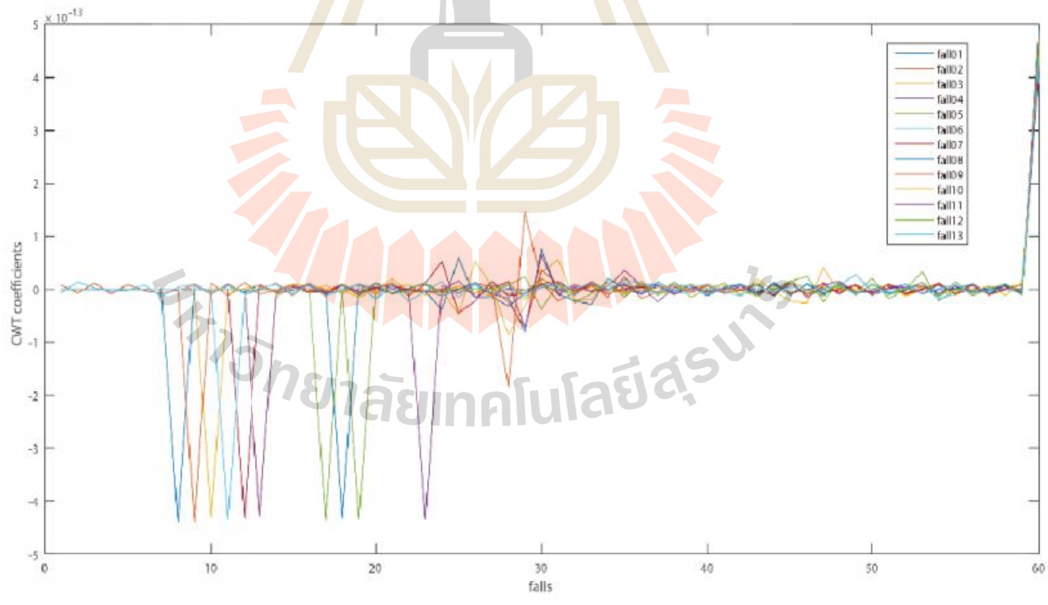


Figure B.9 Falls : SV_{total} after CWT (Haar) coefficients

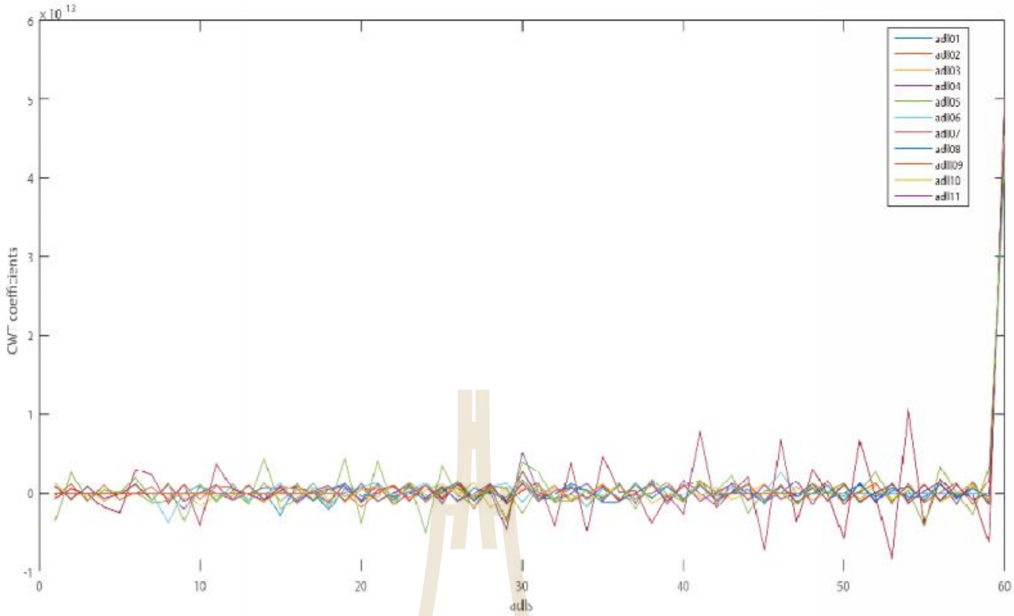


Figure B.10 ADLs : SV_{total} after CWT (Haar) coefficients

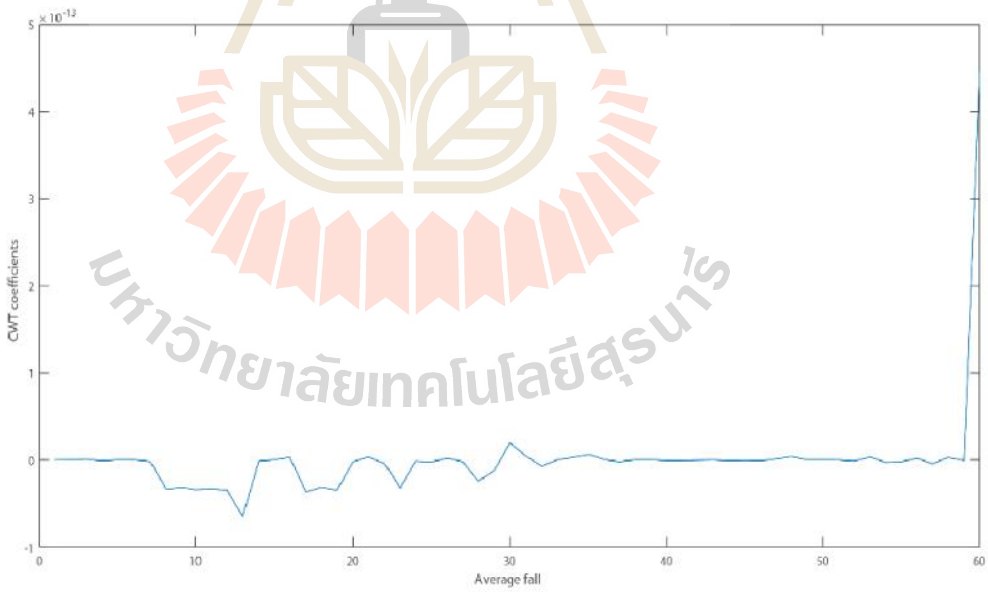


Figure B.11 Average SV_{total} after CWT (Haar) coefficients

APPENDIX C

PUBLICATION

List of Publication

International conference paper

Liang, H., & Usaha, W. (2017, September). Fall detection using lifting wavelet transform and support vector machine. **Computer Science and Information Systems (FedCSIS), 2017 Federated Conference on** (pp. 877-883). IEEE.

Proceedings of the Federated Conference on Computer Science and Information Systems pp. 877–883
DOI: 10.15439/2017F405
ISSN 2300-5963 ACSIS, Vol. 11

Fall Detection using Lifting Wavelet Transform and Support Vector Machine

Hanghan Liang, Wipawee Usaha†
School of Telecommunication Engineering, Suranaree University of Technology, Muang, Nakhon Ratchasima 30000, Thailand
Email: lianghanghan@gmail.com, wusaha@ieee.org†

Abstract—**Frequency domain features of inertial movement enables multi-resolution analysis for fall detection, yet they are computationally intensive. This paper proposes a computationally light frequency domain feature extraction method based on lifting wavelet transform (LWT) which provides computational efficiency suitable for real-time low power devices such as wearable sensors for human fall detection. LWT is combined with support vector machine (SVM) to identify falls from activities of daily living. Performance of the Haar and Biorthogonal 2.2 wavelets were compared with the time domain feature of root-mean square acceleration using a human fall dataset. Results show that the first level detail coefficients features for both Haar and Biorthogonal 2.2 wavelets achieve good overall fall detection accuracy, sensitivity and specificity.**

I. Introduction

As many countries enter the era of aging society, they face critical elderly people's health threats which are fall and related complications caused by the injury [1]. Considering the need of real-time monitoring and ease of use, wearable sensor systems are one of the most promising systems.

Wearable sensor-based fall detection systems, inherently generate continuous monitoring of physiological measurements. Such system is usually a multi-sensor system, comprising sensors such as accelerometers, gyroscopes, pressure sensors and magnetometers. Datasets collected by such wearable sensors are thus, typically multi-dimensional and in large volumes. Such characteristics may cause hinder data processing and fall detection capabilities. Some researches therefore use feature extraction to reduce the amount and the dimensions of data [2] by extracting only necessary features. Existing feature extraction techniques include two main domains, i.e., time and frequency domains. Research such as [1], [3], [4] extracted time domain features including the mean value, maximum value, minimum value and variance, standard deviation of the patient's physiological movements and other special features such as entropy and vertical direction.

In general, time domain features are straightforward and easy to visualize which means light computational burden for feature extraction. So the system is computationally efficient in achieving a real-time fall detection. However, the time domain statistical features considers only the displayed observable trends [2]. Consequently, time domain features may not suffice for accurate fall detection.

Conversely, frequency domain features make use the spectral domain of the collected data which may not be clearly observable in the time domain. Frequency domain features were deployed for fall detection by [5] which used discrete stationary wavelet transform (SWT). In [6], a short time Fourier transform (STFT) was used for human activity recognition, whereby a fall was a subset of data in a series of continuous activities of daily living (ADLs). Ref. [7] created a prototype wavelet of typical fall pattern by using the average acceleration sum vector. The degree of similarity of the signal to the prototype was then computed though wavelet analysis. Results from the same classifier and real-world dataset revealed that the wavelet based features outperformed than other time domain features: upper and lower peak values.

Feature extraction alone only enhance the features of the data acquired by the wearable sensors. However, to detect weather a fall occurred relies on the performance of the detection mechanism. The most common and simplest fall detection is the threshold method [8]. Nevertheless, the performance heavily depends on the fixed threshold level. Hence, it is rarely used alone, and often combined with other machine learning methods such as decision tree (DT) [9], [10], artificial neural networks (ANN) [11], hidden Markov model (HMM) [12] and Support Vector Machine (SVM) [4], [14], [15] can be combined to outperform the threshold method [8], [14]. Among the machine learning methods, SVM was found the most robust for fall detection when compared to other methods such as threshold-based methods and the decision tree method [8]. However, most works which deploy SVM for fall detection use time-series features [8], [16]. It was found that SVM fall detection performance can be improved by a combination of time and frequency domain features [4]. In particular, the discrete Fourier transform

†Corresponding author

This work was financially supported by Suranaree University of Technology under the MOU with Huazhong University of Science and Technology, P.R. China.

(DFT) was used to determine the spectral coefficients which is computationally intensive [4]. On the other hand, the lifting wavelet transform (LWT) is an efficient, light weight frequency domain extraction method [17]. To the best of our knowledge, there is no previous work that has combined LWT with SVM for fall detection. This paper is therefore focused on the study of feature extraction based on LWT used with SVM to detect falls from ADLs using root-mean square value from a single tri-axial acceleration sensor.

The paper is organized as follows. Section II presents the proposed frequency analysis and the support vector machine scheme proposed in the paper. The time domain feature which is used for comparison is also introduced. In section III, the experiment based on a comprehensive fall detection dataset is described. Section IV presents the results and discussion and finally conclusions is given in the final section.

II. Method

A. Frequency domain feature extraction

Feature extraction based on frequency analysis of the body inertia collected from sensors has been studied in the recent literature. Discrete wavelet transform (DWT) has been proposed for mobility monitoring, posture transition and activities classification in [18] using a single chest-mounted sensors. In [19], another frequency domain feature extraction method using short-time Fourier transform (STFT) was proposed to shorten the calculation time of DWT. Despite good results, the short time windows in STFT may not always be suitable for human motion which varies greatly. If windows are too short, STFT may be unable to identify the frequency in such a short period of time. If windows are too large, more information in time domain will lost. If the STFT window size is fixed, STFT may not be suitable for fall detection as human activities are flexible. Unlike DFT in [4], LWT can be constructed from time series signal directly. Unlike DWT in [18], LWT does not require convolution, translation or dilation of traditional mother wavelets. Furthermore, LWT allows in place calculation, with no need for auxiliary memory. Therefore, LWT provides computational efficiency suitable for real-time low power devices such as wearable sensors. In the following subsection, we describe LWT in more details.

B. Lifting Wavelet Transform

LWT has been introduced by Sweldens in 1997 [17]. The scheme theory is often described as three steps: split, predict and update. The split step is to split a signal into to two independent sequences, i.e, the even half and odd half sequences. Let x_i be the original discrete signal at time index i. Let $even_i$ (odd_i) denote the i^{th} index of the even (odd) sequence. We have that $even_i = x_{2i}$ and $odd_i = x_{2i+1}, i \in I$.

LWT is a recursive algorithm which splits the signal into halves at each level. If the original signal has 2^n elements, then the next level will operate on $2^{(n-1)}$ elements. Hence, if the original signal has 256 elements, there will be 8 levels with the next level having 128 elements. The subsequent levels will have 64, 32, 16, 8, 4, 2 and 1 element. The odd values in the next level $j+1$ is predicted from the even value at level j:

$$cD_{j+1,i} = odd_{j,i} - P(\,even_{j,i}\,) \qquad (1)$$

where P is the predict function which approximates the signal, cD is the high frequency component of x_i. This is called the *Predict* phase. The even values at the next level can be found from

$$cA_{j+1,i} = even_{j,i} + U\,(\,cD_{j+1,i}\,) \qquad (2)$$

where U is the update operation that updates on the differences from the odd values, cA is the low frequency component of x_i. This is called the *Update* phase. The multi-level lifting scheme can be summarized in Fig. 1. The averages are sometimes called approximate coefficients whereas the differences are called the detail coefficients. There are two types of wavelets used in this paper.

1) Haar wavelet:

Predict :

$$cD_{j+1,i} = odd_{j,i} - even_{j,i} \qquad (3)$$

Update :

$$cA_{j+1,i} = even_{j,i} + \frac{1}{2} cD_{j+1,i} \qquad (4)$$

2) Biorthogonal 2.2 wavelet:

Predict :

$$cD_{j+1,i} = odd_{j,i} - \frac{1}{2}(even_{j,i} + even_{j,i+1}\,) \qquad (5)$$

Update :

$$cA_{j+1,i} = even_{j,i} + \frac{1}{4}(cD_{j+1,i-1} + cD_{j+1,i}) \qquad (6)$$

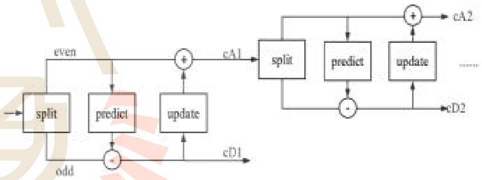


Fig. 1 Forward lifting scheme

Figure 2 shows a sample fall plot of the original signal and the *first* level LWT coefficient. The number of coefficients of cA1 (average or low frequency part) and cD1 (detail or high frequency part) are half of the original signal according to the number of data points. By comparing cA1, cD1 and the root-mean square acceleration (SV_{total}) in Fig. 2, it is seen that cA1 greatly correlates with the original signal. Note that cD1 also shows a peak similar to the original signal signifying a fall which occurred during the red highlighted window of one second. However, the baseline zero illustrated a more distinguished fall feature than cA1. Therefore, cD1 was preferable than cA1 for feature extraction of falls.

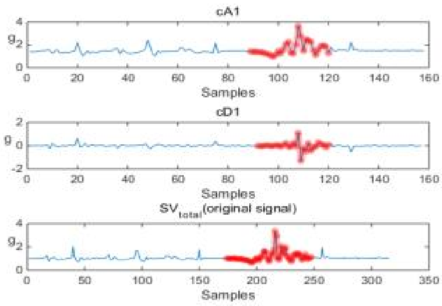


Fig. 2 A sample fall plot of SV_{total} and after LWT cD1 and cA1 with data points inside "fall" window highlighted

C. Time domain feature

The tri-axial acceleration data collected contains A_x, A_z, A_y in x-axis, z-axis and y-axis as a function of time. All accelerometer data were in factors of gravity units (g). The accelerometer components were used to calculate the root-mean square acceleration denoted by total sum vector SV_{total}:

$$SV_{total} = \sqrt{{A_x}^2 + {A_y}^2 + {A_z}^2} \quad . \tag{7}$$

D. Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning model which is commonly used for anomaly detection and classification [20], [21], [22]. As a supervised learning model, SVM requires training from datasets with "labels." The SVM concept is to map a set of data points from the real-world to a higher dimensional space. A boundary or hyperplane is created in a high dimensional space by training datasets to classify the features into fall or non-fall. Since the fall detection system inherently generates long-term continuous monitoring of physiological measurements, such datasets are usually large. Such characteristic may cause difficulty in data processing. To reduce the amount of data and achieve a higher calculating speed, the features of the data may be extracted from these raw datasets.

To train the SVM, the data points in the dataset must be labeled. For example, in time domain, SV_{total} was directly used as input feature. We labeled all the ADLs data points with "-1" whereas falls were labeled "1." Fig. 3 depicts a sample plot of a fall along with non-fall activities like walking around and lying on the ground. Point A shows the peak value of the dataset. A highlighted window size with point A placed at the middle of the window is constructed. Within such window, all the data points are labeled "1" and the remaining data points outside this window are labeled "-1." The goal of SVM is therefore to distinguish the labels among the tested datasets using the model obtained from the trained data. The data points are typically non-linearly separable to classify in low dimensional space. However, if these points are projected onto a higher dimensional space, it is possible to find a hyperplane to classify the labels. Such projection is obtained through use of kernel functions such as linear, polynomial, sigmoidal, or the Gaussian radial base functions. It is with this kernel trick that makes SVM a powerful model to classify the labels in higher dimensional space. In the next section, the experiment settings are presented.

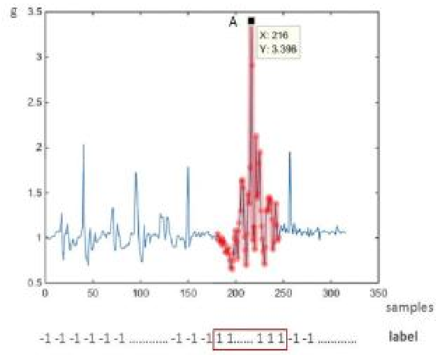


Fig. 3 A sample plot of fall SV_{total} with data points inside window highlighted

III. Experiment

As mentioned in the previous section, SVM requires training labeled datasets. As data input in the fall detection scenario involves both non-falls and falls data, We trained with both falls and non-falls italic in Table I.we first evaluate the SVM model with a hybrid fall and non-fall activities. The objective is to evaluate a suitable training dataset for SVM to detect falls. For the sake of simplicity, only the time domain feature (SV_{total}) is studied.

Once a SVM model is trained, we proceed to study the comparison between features in the time domain (SV_{total}) and frequency domain (LWT using Haar and Biorgthogonal 2.2 wavelets). Note that there are existing works which combined features in both time domain and frequency domain of data, the type of sensors, the number and position of sensors on human body, and in the volume of dataset for training and testing [1], [5]. From results gathered from existing literature, we focus on data collected from a single tri-axial acceleration sensor due to its low cost, reliability and efficiency.

A. Performance metrics

To evaluate the performance, we measure the True Positives (TP) or True Negatives (TN) which refers to the number of events correctly identified or correctly rejected. False Positives (FP) or False Negatives (FN) which represent the number of events incorrectly identified or incorrectly rejected [23]. These measurements provide the following necessary metrics required to evaluate the fall detection method:

1) *Sensitivity (SE)* or true positive rate is the capability to detect a fall correctly. It is an indicator to judge whether a system will miss a fall. It is given by

$$SE = \frac{TP}{TP+FN} \times 100\% \tag{8}$$

2) *Specificity (SP)* or true negative rate is the ability to detect a fall only if a fall really occurred. It is to avoid false alarm given by

$$SP = \frac{TN}{TN+FP} \times 100\% \tag{9}$$

3) *Accuracy (AC)* or correct rate refers to the overall freedom from false. This is given by

$$AC = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \tag{10}$$

It is worth noting that SVM classifies data points individually. However, to detect a fall within a certain window as shown in Fig. 3, a set of data points must be classified rather than just a single data point. Therefore, to determine a suitable decision region to decide whether a fall has occurred, we use a simple calculation for the percentage of predicted fall label "1" over the number of labels observed in an activity to compare with a predetermined threshold:

$$T = \frac{\textit{the number of predicted "1"}}{\textit{the number of testing data points}} \tag{11}$$

If T > threshold, the activity is a fall. Otherwise, else it is a non-fall activity.

B. Training SVM Model

We hypothesize that the best type of training dataset will be the combined set of both fall and ADLs dataset. Since not only falls but also ADLs data are contained in the hybrid training dataset, the more comprehensive information contained in training dataset, the more likely the model will decide correctly.

The dataset we used to train and test the SVM models have been obtained from [24] including 70 activities (tri-axial acceleration of 30 falls and 40 non-falls collected and video recorded with Kinect camera) with details given in Table I. The tri-axial accelerometer data was sampled at 60Hz. Therefore, a one-second window for fall detection consists of

TABLE I.
DATASETS USED IN EXPERIMENT[1]

Data file	Activities description	Data file	Activities description
	Falls Activities		
fall-01-acc	*From vertical falling left on the floor*	**fall-16-acc**	From sitting falling right on the floor
fall-02-acc	*From sitting falling left on the floor*	**fall-17-acc**	From vertical falling forward on the floor
fall-03-acc	From vertical falling left on the floor	**fall-18-acc**	From sitting falling left on the floor
fall-04-acc	From sitting falling left on the floor	**fall-19-acc**	From vertical falling right on the floor
fall-05-acc	*From vertical falling right on the floor*	**fall-20-acc**	From sitting falling right on the floor
fall-06-acc	*From sitting falling right on the floor*	**fall-21-acc**	From vertical falling right on the floor
fall-07-acc	From vertical falling left on the floor	**fall-22-acc**	From sitting falling left on the floor
fall-08-acc	From sitting falling right on the floor	**fall-23-acc**	From vertical falling right on the floor
fall-09-acc	From vertical falling left on the floor	**fall-24-acc**	From sitting falling left on the floor
fall-10-acc	From sitting falling left on the floor	**fall-25-acc**	From vertical falling forward on the floor
fall-11-acc	From vertical falling right on the floor	***fall-26-acc***	*From sitting falling forward on the floor*
fall-12-acc	From sitting falling right on the floor	**fall-27-acc**	From vertical falling forward on the floor
fall-13-acc	*From vertical falling forward on the floor*	**fall-28-acc**	From sitting falling forward on the floor
fall-14-acc	From sitting falling right on the floor	**fall-29-acc**	From vertical falling forward on the floor
fall-15-acc	From vertical falling forward on the floor	**fall-30-acc**	From sitting falling forward on the floor
	Non-falls Activities (ADLs)		
Data file	**Activities description**	**Data file**	**Activities description**
adl-01-acc	*Walking, then squatting*	**adl-21-acc**	From vertical lying on the bed
adl-02-acc	Walking, then squatting	**adl-22-acc**	From vertical lying on the bed
adl-03-acc	Walking, then squatting	**adl-23-acc**	From vertical lying on the bed
adl-04-acc	*Bending 90 degree to pick up something*	**adl-24-acc**	Walking, then squatting
adl-05-acc	Squatting to pick up something	**adl-25-acc**	From vertical to sitting onto a chair
adl-06-acc	Squatting to pick up something	**adl-26-acc**	Walking, then squatting
adl-07-acc	*From vertical to sitting onto a chair*	**adl-27-acc**	From vertical to sitting onto a chair
adl-08-acc	From vertical to sitting onto a chair	**adl-28-acc**	Walking, then squatting
adl-09-acc	From vertical to sitting onto a bed	**adl-29-acc**	From vertical to sitting onto a chair
adl-10-acc	*From vertical lying on the bed*	**adl-30-acc**	From vertical lying leftward on the ground
adl-11-acc	From vertical lying rightward on the bed	***adl-31-acc***	*From vertical lying forward on the ground*
adl-12-acc	Walking, then squatting	**adl-32-acc**	From vertical lying forward on the ground
adl-13-acc	Walking, then squatting	**adl-33-acc**	From vertical lying forward on the ground
adl-14-acc	Walking, then squatting	**adl-34-acc**	From vertical lying forward on the ground
adl-15-acc	Bending 90 degree to pick up something	**adl-35-acc**	From vertical lying forward on the ground
adl-16-acc	Bending 90 degree to pick up something	**adl-36-acc**	From vertical lying rightward on the ground
adl-17-acc	*Squatting to pick up something*	**adl-37-acc**	From vertical lying rightward on the ground
adl-18-acc	From vertical to sitting onto a bed	**adl-38-acc**	From vertical lying forward on the ground
adl-19-acc	From vertical to sitting onto a chair	**adl-39-acc**	From vertical lying forward on the ground
adl-20-acc	From vertical to sitting onto a bed	**adl-40-acc**	From vertical lying forward on the ground

Source: http://fenix.univ.rzeszow.pl/~mkepski/ds/uf.html
[1] The italic activities were used as training dataset.

60 data points. The dataset was divided into training set and testing set based on activities in the matching video of each data file. Table I consists of fall and non-fall (ADLs) activities. The SVM model has been trained with the datasets obtained in italics in Table I for a comprehensive dataset of various falls and ADL activities.

Once the data points are labeled and trained, the SVM model is obtained. The SVM model is then used to classify the testing data. The dataset remaining (non-italic activities) in Table I are used for testing. For each dataset tested, a data point is labeled "1" for data points predicted as a fall data point, or "-1" for data points predicted as non-fall data point. If the ratio of fall labels in an activity exceeds the determined threshold, then a fall has been detected. For each tested dataset, TP, TN, FP and FN is measured for the calculation of SE, SP and AC to evaluate the SVM model. Results are presented in Section IV.

C. Comparing Time and Frequency domain features

This part of the experiment is to compare the time domain feature (based on SV_{total}) and the frequency domain features (based on Haar and Bioorthogonal 2.2 wavelets). Using the SVM model obtained in the previous experiment, a suitable level threshold level to detect a fall event for each feature is then found. For each feature, the percentage levels of threshold is tested at 10%, 20%, 30%, 40% and 50%. Then level is tested at finer threshold values. Results are shown in Table II.

We then investigate closely how multiple levels of LWT coefficients affect the fall detection performance by evaluating the first five levels of coefficients of the Haar and Biorthogonal 2.2 wavelets. Results are shown in Table III.

IV. Results and Discussion

A. Training SVM Model

Results show that the SVM model trained and tested with *time domain* datasets of both falls and ADL activities gave a 100% sensitivity, 97.14% of specificity and 98.31% accuracy. It should be noted that the 100% sensitivity is obtained from *offline* datasets with a predetermined threshold found from observing these datasets. Furthermore, a larger dataset collected from online simulated falls is currently under investigation.

B. Comparing Time and Frequency domain feature

Table II shows the performance comparison between time and frequency domain features at different levels of thresholds.

1) *Root-mean square acceleration:* Table II shows that the best threshold for the time domain feature should be between 10% to 20%. With fine threshold tuning, it is found that a threshold of 17-18% showed better preference than others (shown in bold fonts). Therefore, we chose 17% as the threshold to classify a fall or non-fall for time domain feature.

2) *LWT with Haar Wavelet:* The appropriate threshold for Haar LWT is found by also ranged from 10% to 50%. As shown in Table II, the best achieved threshold should be under 10%. To fine tune the threshold levels, the threshold is varied from 2% to 10%. It is found that the threshold at 8% outperformed other levels (shown in bold fonts). Thus, we chose 8% as the threshold for LWT using Haar wavelet. In Table III, multiple levels of LWT coefficients (cD1 to cD5) are evaluated. When tested with ADLs & Falls dataset, all specificity, specificity and accuracy values of 100% was achieved only in cD1 (shown in bold fonts). This result indicated that Haar LWT CD1 coefficients achieved a goal such that no ADL has been misclassified as a fall and detected most of the falls when training and testing using finite activities in Table I

3) *LWT with Biorthogonal 2.2 Wavelet:* From Table II, the optimal threshold for Biorthogonal 2.2 (Bior 2.2) should be under 10%. With a finer threshold search, results indicate that threshold level of 6% is the best level with 100% sensitivity, specificity and accuracy *(shown in bold fonts).* Similar to Haar LWT, Bior 2.2 LWT coefficients also show a good performance distinguishing falls from ADLs when using most cD levels. In Table III, cD1 also outperformed other levels of coefficients similar to Haar wavelet *(shown in bold fonts)*. The reason may be the information contained in the frequency components that is helpful to classify activities by SVM. The cD1 components contained the most distinguishable information of falls, while cD5 contained the least information. Generally, Haar was slightly better at distinguishing ADLs from falls than Bior 2.2, whereas both LWT features outperform the time domain feature of root-mean square acceleration alone. It is worth noting that these results are obtained by a comprehensive human fall dataset with video captures obtained from [24] which allow the thresholds and detail coefficients to be predetermined offline. Current ongoing work involves implementing the LWT and SVM on actual wearable sensor devices to be evaluated online for human fall detection for accuracy and efficiency.

V. Conclusions

In this paper, we propose a computationally light frequency domain feature extraction method called lifting wavelet transform (LWT) for a wearable sensor human fall detection device combined with a fall identifier using support vector machine model. The performance of the LWT using Haar and Biorthogonal 2.2 wavelets, together with the time domain feature of root-mean square acceleration have been evaluated with raw dataset acquired from a single tri-axial acceleration sensor from an existing human fall and activities of daily living dataset.

Based on the dataset, suitable thresholds and level of detail coefficients can be predetermined. Consequently, the LWT frequency domain features are shown to have better performance than time domain features in terms of sensitivity, specificity and accuracy. Given a one-second window size under a sampling frequency of 60Hz, the best threshold in terms of the percentage of fall labels ("1") per window is as follows, 18% for the time domain feature using the root-mean square acceleration, and 8% for Haar and 6% for

Biorthogonal 2.2 LWT wavelets when the SVM model is trained with both fall and non-fall datasets. The frequency domain feature from cD1 for both Haar and Biorthogonal 2.2 wavelets achieved 100% overall accuracy whereas 98.31% overall accuracy was attained for the time domain feature, SV_{total}. All features achieved 100% sensitivity from this dataset. In terms of specificity, the time domain feature, SV_{total}, attained up to 97.14% whereas the two LWT features attained 100%. In a final note, ongoing work involves implementing the LWT and SVM on actual wearable sensor devices to be evaluated for human fall detection accuracy and reliability in real-time.

ACKNOWLEDGMENT

The authors would like to thank Suranaree University of Technology for the financial support under the MoU with Huazhong University of Science and Technology, P.R. China for Ms.Hanghan Liang to conduct this research.

REFERENCES

[1] Pierleoni P, Pernini L, Belli A, et al. "SVM-based fall detection method for elderly people using Android low-cost smartphones," *IEEE Sensors Applications Symposium (SAS 2015)*, 2015: 1-5.

[2] Banaee H, Ahmed M U, Loutfi A., "Data mining for wearable sensors in health monitoring systems: a review of recent trends and challenges," *Sensors*, 2013, 13(12): 17472-17500.

[3] Carlsson T., "*Individualized Motion Monitoring by Wearable Sensor: Pre-impact fall detection using SVM and sensor fusion*," Masters Thesis, School of Technology and Health, KTH Royal Institute of Technology, Stockholm, Sweden, 2015.

[4] Özdemir A T, Barshan B., "Detecting falls with wearable sensors using machine learning techniques," *Sensors*, 2014, 14(6): 10691-10708.

[5] Su B Y, Ho K C, Rantz M J, et al., "Doppler radar fall activity detection using the wavelet transform," *IEEE Transactions on Biomedical Engineering*, 2015, 62(3): 865-875.

[6] Björklund S, Petersson H, Hendeby G. , "Features for micro-Doppler based activity classification," *IET Radar, Sonar & Navigation*, 2015, 9(9): 1181-1187.

[7] Palmerini L, Bagalà F, Zanetti A, et al., "A wavelet-based approach to fall detection," *Sensors*, 2015, 15(5): 11575-11586.

[8] Aziz O, Musngi M, Park E J, et al., "A comparison of accuracy of fall detection algorithms (threshold-based vs. machine learning) using waist-mounted tri-axial accelerometer signals from a comprehensive set of falls and non-fall trials," *Medical & Biological Engineering & Computing*, 2017, 55(1): 45-55.

[9] Bilski P, Mazurek P, Wagner J, et al., "Application of Decision trees to the Fall Detection of elderly People using Depth-based sensors," *Proc. IEEE International Conference on Intelligent Data Acqut and Advanced Computing Systems (IDAACS 2015)*, Warsaw, Poland, September, 2015.

[10] Parkka J, Ermes M, Korpipaa P, et al. , "Activity classification using realistic data from wearable sensors," *IEEE Transactions on Information Technology in Biomedicine*, 2006, 10(1): 119-128.

[11] Wang Z, Jiang M, Hu Y, et al., "An incremental learning method based on probabilistic neural networks and adjustable fuzzy clustering for human activity recognition by using wearable sensors," *IEEE Transactions on Information Technology in Biomedicine*, 2012, 16(4): 691-699.

TABLE II.
PERFORMANCE COMPARISON[1] OF DIFFERENT THRESHOLDS FOR TIME AND FREQUENCY DOMAIN FEATURES

	INITIALLY ESTIMATED THRESHOLDS					**FINE TUNED THRESHOLDS[2]**			
	TIME DOMAIN (SV_{TOTAL})								
THRESHOLD	10%	20%	30%	40%	50%	15%	**17%**	**18%**	19%
SE (%)	100	95.83	91.67	87.50	87.50	100	**100**	**100**	100
SP (%)	80	100	100	100	100	94.29	**97.14**	**97.14**	91.43
AC (%)	88.14	98.31	96.61	94.92	94.92	96.61	**98.31**	**98.31**	98.31
	FREQUENCY DOMAIN (HAAR, CD1)								
THRESHOLD	10%	20%	30%	40%	50%	2%	4%	6%	**8%**
SE (%)	95.83	87.50	87.50	66.67	33.33	100	100	100	**100**
SP (%)	100	100	100	100	100	82.35	85.29	97.06	**100**
AC (%)	98.28	94.83	94.83	86.21	72.41	89.66	91.38	98.28	**100**
	FREQUENCY DOMAIN (BIOR2.2, CD1)								
THRESHOLD	10%	20%	30%	40%	50%	4%	5%	**6%**	
SE (%)	95.83	87.50	83.33	62.50	33.33	100	100	**100**	
SP (%)	100	100	100	100	100	88.57	97.14	**100**	
AC (%)	98.31	94.92	93.22	84.75	72.88	93.22	98.31	**100**	

[1]Trained with ADLs & Falls SVM Model and tested by ADLs & Falls dataset
[2] Bold fonts indicate the best performance for each feature

TABLE III.
PERFORMANCE COMPARISON[1] OF DIFFERENT FREQUENCY DOMAIN COMPONENTS[2]

FEATURES	**CD1**	**CD2**	**CD3**	**CD4**	**CD5**
			HAAR		
SE (%)	**100**	83.33	95.83	87.50	95.83
SP (%)	**100**	100	100	100	100
AC (%)	**100**	93.10	98.28	94.83	98.28
			BIOR2.2		
SE (%)	**100**	100	91.67	100	100
SP (%)	**100**	94.29	100	82.86	5.71
AC (%)	**100**	96.61	96.61	89.83	44.07

[1]Trained with SVM Model-3 and tested by ADLs &Falls dataset
[2] Bold fonts indicate the best performance for each feature

[12] Tong L, Song Q, Ge Y, et al., "HMM-based human fall detection and prediction method using tri-axial accelerometer," IEEE Sensors Journal, 2013, 13(5): 1849-1856.
[13] Kianoush S, Savazzi S, Vicentini F, et al., "Leveraging RF signals for human sensing: fall detection and localization in human-machine shared workspaces," //Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on. IEEE, 2015: 1456-1462.
[14] Pierleoni P, Belli A, Palma L, et al. "A high reliability wearable device for elderly fall detection," IEEE Sensors Journal, 2015, 15(8): 4544-4553.
[15] Liu S H, Cheng W C., " Fall detection with the support vector machine during scripted and continuous unscripted activities," Sensors, 2012, 12(9): 12301-12316.
[16] Shibuya N, Nukala B T, Rodriguez A I, et al., "A real-time fall detection system using a wearable gait analysis sensor and a support vector machine (SVM) classifier," Mobile Computing and Ubiquitous Networking (ICMU), 2015 Eighth International Conference on. IEEE, 2015: 66-67.
[17] Sweldens, Wim. "The Lifting Scheme: A Construction of Second Generation Wavelets," Journal on Mathematical Analysis. SIAM. 1997, 29 (2): 511–546.
[18] Wójtowicz B, Dobrowolski A, Tomczykiewicz K., "Fall detector using discrete wavelet decomposition and SVM classifier," Metrol. Meas. Syst, 2015, 22(2): 303-314.
[19] Isu Shin, Jongsang Son, Soonjae Ahn, Jeseong Ryu, Sunwoo Park, Jongman Kim, Baekdong Cha, Eunkyoung Choi, and Youngho Kim, "A Novel Short-Time Fourier Transform-Based Fall Detection Algorithm Using 3-Axis Accelerations," Mathematical Problems in Engineering, 2015, Article ID 394340.
[20] Hsu C W, Chang C C, Lin C J. A practical guide to support vector classification, 2003.
[21] Cortes C, Vapnik V., "Support-vector networks," Machine Learning, 1995, 20(3): 273-297.
[22] Chang C C, Lin C J., " LIBSVM: a library for support vector machines," ACM Transactions on Intelligent Systems and Technology, 2011, 2(3): 27.
[23] Altman D G, Bland J M., "Diagnostic Tests : Sensitivity and specificity," British Medical Journal, 1994, 308(6943): 1552.
[24] Kwolek B, Kepski M., "Human fall detection on embedded platform using depth maps and wireless accelerometer," Computer Methods and Programs in Biomedicine, 2014, 117(3): 489-501.

BIOGRAPHY

Liang Hanghan was born on January 20th, 1992 in Hubei, China. She finished her high school from Shashi High School, Jingzhou, Hubei. In 2015, she got her Bachelor's Degree in School of Telecommunication Engineering, Institute of Electronic Information Engineering, Huazhong University of Science and Technology, Wuhan, China. After she got her Bachelor diploma, she continued to study as a post graduate student in School of Telecommunication Engineering, Institute of Engineering, Suranaree University of Technology, Nakhon Ratchasima, Thailand.