# STRIDE LENGTH ESTIMATION USING ANN

A THESIS SUBMITTED TO AUCKLAND UNIVERSITY OF TECHNOLOGY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF COMPUTER AND INFORMATION SCIENCES

July 2018

By

YU LIU

School of Engineering, Computer and Mathematical Sciences

# Abstract

Measuring the stride length of a pedestrian is an essential task for many applications, such as augmented reality (AR) applications tracking devices and motion monitoring devices. In the traditional position estimation method such as the double integral of the acceleration signal, the signal noise of the acceleration from the Inertial measurement units (IMU) leads to a cumulative error. It is the bottleneck of the positioning performance in most of Inertial Navigation System (INS). Moreover, in many applications, such as AR and tracking devices, sensors are attached to the human body and used in the indoor environment, which is hard to obtain the position of the device through GPS or traditional INS. In this paper, we proposed a novel method that transforms the position information to step length, and then we use Artificial Neural Network (ANN) to estimate the step length through the data from the IMU. The conducted experiments show that the proposed method achieved less than $2\%$ error in a distance of $62.3m$. Comparing to the traditional double integral method, it has superior performance and better ability to handle the signal noise from a low-cost IMU.

**Keywords: Neural Network, IMU, Stride Length Estimation**

# Contents

# List of Tables

# List of Figures

# Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.

Signature of student

# Acknowledgements

Firstly,I would like to thank my supervisor, A/Prof, Loulin Huang for the support of my master thesis, for his patience, motivation and his guidance through my research.

Secondly, I would also like to thank Jian Huang, the senior technician in AUT, who help me assemble the hardware for the experiments.

Thirdly, I am grateful for Wuxin Yang who assisted me in MATLAB coding and the collection of the data.

Last but not least, I would give my appreciation to the Department of Computer Science and Department of Mechanical Engineering of AUT computer for their help.

# Chapter 1

# Introduction

## 1.1 Research Background

To measure stride length of a pedestrian is necessary in many applications including localization of an augmented reality (AR) device through the position of the operator (Mulloni, Seichter & Schmalstieg, 2011; Ahn & Han, 2012), tracking of firefighter in search and rescue mission (Faulkner, Alwood, Taylor & Bohlin, 2010; Foxlin, 2005) and monitoring the motions of people required in the situations in health care or training etc (Stone & Skubic, 2011; Trojaniello, Ravaschio, Hausdorff & Cereatti, 2015).

For the outdoor pedestrians positioning, Satellite Position System (SPS) is the most common technologies used. One of the well-known SPS application is Global Position System (GPS). There are 24 to 32 earth orbit satellites which transmit radio signals for GPS. Based on these signals, GPS receivers can determine the location, time and velocity (Grewal, Weill & Andrews, 2007) of the carrier. It not only works in most weather condition and location, but also provides satisfactory accuracy of the navigation. What's more, it is cheaper compared to other navigational systems. The cost of a GPS chip is under $5 US dollar (Colwell, 2007). In this way, GPS is widely used in various application such as car navigation, mobile orientation, autonomous vehicle, Unmanned

Aerial Vehicle (UAV) and performance analysis. However, GPS chip is power hungry, it can drain battery in 8 to 12 hours (Colwell, 2007). GPS signals can be affected by many atmospheric conditions (i.e. geomagnetic storms) and electromagnetic interference. This leads to an error of about 5 to 10 meters in positioning accuracy (Terrier, Ladetto, Merminod & Schutz, 2000). Furthermore, GPS signals can be blocked by obstacles such as terrain, buildings and water (Grewal et al., 2007). Therefore, GPS is not able be used indoor, underwater or in dense tree regions (Chiang, Chang, Li & Huang, 2009). Many AR applications (such as games) are designed for the indoor environment. Without a capability of tracking the position of the device indoor, the application of AR will be limited. One popular solution is using Inertial navigation system (INS) which is the most common technology used for indoor positioning (Chiang et al., 2009). The INS uses the data obtained by the IMU (Inertial Measurement Unit) to estimate the position of the object(Kong, 2004). An IMU usually contains a cluster of sensors such as three axis-installed accelerometers, three axis-installed gyroscope and magnetometer (Mourcou, Fleury, Franco, Klopcic & Vuillerme, 2015). In INS, the position can be obtained by from its the displacement from a known starting point (Levi & Judd, 1996). The displacement is derived from the integration of velocity which in terms is got from the integration of the acceleration with time. The process of a standard INS is shown in Figure 1.1.

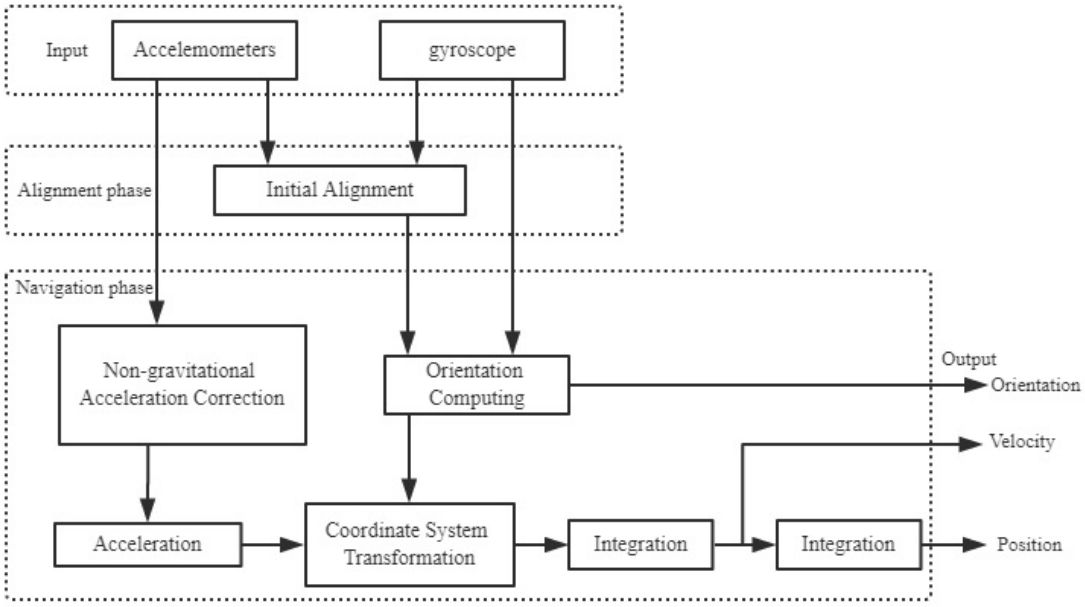



Figure 1.1: Process of INS

Because of the nature of the human gait, a pedestrian's travelled distance consists of strides. Double integration is a common method to estimate the distance. Generally, the recorded acceleration data collected by the IMU contains significant noises, which will lead to cumulative error in each step after double integration. Moreover, the magnitude of the noise is enlarged by the square of time (Levi & Judd, 1996). To solve this problem is the main motivation of this dissertation.

## 1.2 Research objective

Artificial neural network (ANN) is considered as a powerful tool to handle the linear and nonlinear fitting and classification problems. It has been applied in many fields such as medical diagnosis (myocardial infarction (Baxt, 1991), tumour prediction (Khan et al., 2001) and seizure detection (Ahmad, Saeed, Saleem & Kamboh, 2016; Shoeb, 2009)), atmospheric sciences (Gardner & Dorling, 1998), object recognition (He, Lau, Liu, Huang & Yang, 2015; Pohtongkam & Srinonchat, 2016) and trajectory prediction (Payeur, Le-Huy & Gosselin, 1995; Meireles, Almeida & Simões, 2003). In this dissertation, we use ANN to estimate the stride length. Firstly, a person walks on the floor in a normal gait. Then the stride length of each step measured by hand is used as a target data for the ANN. Secondly, the IMU is put on the boots of the same person. It measures the accelerations and Euler angles corresponding to the person's walk. Thirdly, the collected data will be smoothed and filtered. Then the feature points will be extracted from the processed data and be used as the input data for ANN training. At last, the trained ANN will be applied to estimate the stride length and the result will be analyzed.

## 1.3 Organization of the Dissertation

This dissertation is organized as follow. Chapter 2 is on the literature review of the traditional gait detection method, step length estimation technique. Chapter 3 is on the theory that related to the experiments such as artificial neural network, noise filtering method and coordinate systems. Chapter 4 is on the description of the methodology, the equipment and data collection of the experiments. Chapter 5 is on the steps to analyze and process the collected data in the experiments. In Chapter 6, the experimental results are presented. In Chapter 7, the conclusions are given and the future works are discussed.

# Chapter 2

# Literature Review

## 2.1 Detection Of The Gait Cycle

In our method, the first task of stride length estimation is to extract the steps from the data collected. The human bipedal movement can be divided into four steps: stance, heel off, swing and heel strike (Willemsen, Bloemhof & Boom, 1990; R. G. Stirling, 2004). In a simpler form, Everett and Kell (2010) simplified the human bipedal movement into two phases: stand phase and swing phase, as shown in Figure 2.1.

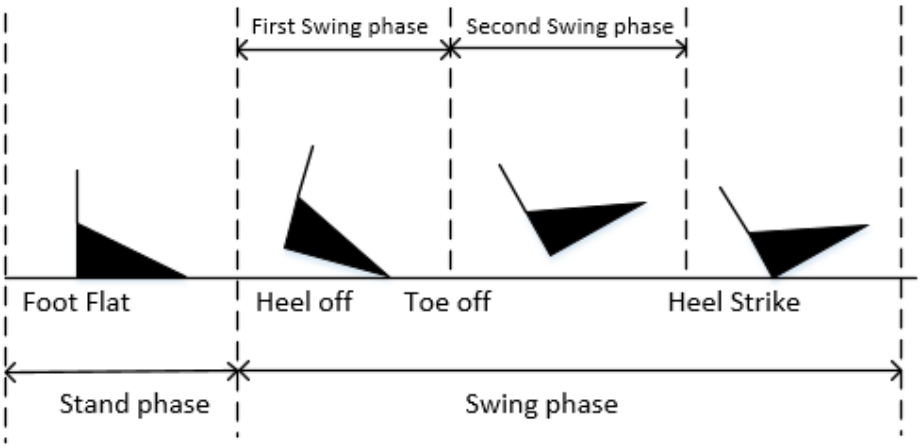



Figure 2.1: Illustration of gait cycle

When the foot is planted on the ground, the sensor that attached to the foot will remain stationary during the stand phase. The data collected by the inertial sensor will remain a stable value without large fluctuations. Therefore, the threshold of fluctuations of the data (e.g. from an accelerometer) can be identified to detect the stand phase (R. Stirling, Collin, Fyfe & Lachapelle, 2003; Castaneda & Lamy-Perbal, 2010). Also, Ojeda and Borenstein (2007), Woodman and Harle (2008) used angular velocity thresholds and Foxlin (2005) used the thresholds of both acceleration data and angular velocity data to detect the stand phase. Moreover, the threshold of magnetometer data can also assist the stance phase detection(Jimenez, Seco, Prieto & Guevara, 2009).

Jimenez et al. (2009) reported the stand phase detection error rate of 0.1% and 0.2% with the thresholding method on accelerometer data and angular velocity respectively. However, this method requires the sensor mounted to the foot which is affected by the deformation and bounce of the footwear (Kim, Jang, Hwang & Park, 2004).

There are other techniques such as peak detection, zero crossing detection, auto-correlation and spectral analysis developed to identify the specific data segmentation as listed below.

- Peak detection — In this method, gait is detected by hell strikes which are related to sharp changes in vertical acceleration. However, the large forces cause the sensor bounce which produces multiple peaks can cause errors in the gait detection (Fang et al., 2005; Ying et al., 2007).
- Zero crossings — A cheaper way to monitor zero-cross acceleration values. This method is using the changing rate of the velocity data to determine the stance phase (Shin, Park, Kim, Hong & Lee, 2007).
- Auto-correlation — Regardless of the sensor attachment position, the gait cycle can result in the strong periodicity of the collected data. The period can be extracted by searching the maximum value in mean-adjusted auto-correlation of

the sensor data series (Ying et al., 2007). Whether the peak value corresponds to stride depends on the mounted position of the sensor. In other words, when a sample data series of a certain stride is previously collected, the same process can repeatedly be used to identify the stride if the data is cross-correlation with the previous record. The auto-correlation method depends on detecting the cyclicity of the sensor signals. However, it is hard to implement this algorithm to handle variations in walking speed.

Table 2.1: Gait cycle detection summary

| Technique | Author | Site | Signals used |
|---|---|---|---|
| Thresholding | R. Stirling et al. (2003) | Foot | Acceleration, magnitude |
| | Castaneda & Lamy-Perbal (2010) | Foot | Magnitude |
| | Ojeda & Borenstein (2007) | Foot | Gyroscope, magnitude |
| | Woodman & Harle (2008) | Foot | Magnitude, angular velocity, acceleration |
| | Foxlin (2005) | Foot | Gyroscope, acceleration |
| | Jimenez et al. (2009) | Foot | Magnitude, compass |
| Peak detect | Fang et al. (2005) | Both feet | Acceleration |
| | Ying et al. (2007) | Waist | Acceleration, compass |
| Zero-crossing | Shin et al. (2007) | Body | Acceleratio, gyroscope |
| | Goyal et al. (2011) | Waist | Acceleratio |
| Auto-correlation | Ying et al. (2007) | Waist | Acceleration, compass |
| | Ria et al. (2012) | Pocket | Not specified |

## 2.2 Stride Length Estimation

According to Godha, Lachapelle and Cannon (2006), the distance between the initial contact point of one foot $(s_1)$ and the initial contact point of the "opposite" foot $(s_2)$ is called the stride length. However, in many experiments, as the sensor is attached on foot, the stride length is usually defined as the distance between the constant initial

contact points of the same foot $(s)$.

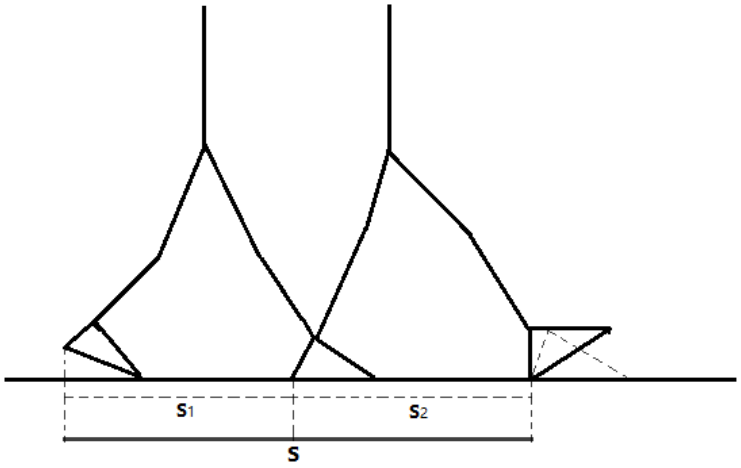



Figure 2.2: The illustration of the stride length

### 2.2.1 Integration of acceleration

One popular method of estimating the stride length is the double integration of the signal (acceleration) from the accelerometer. Numerical integration methods like modified trapezoidal rule and customized Simpson's rule are usually used (Bamberg, Benbasat, Scarborough, Krebs & Paradiso, 2008; Truong, Lee, Kwon & Jeong, 2016; Rampp et al., 2015; Bachschmidt, Harris & Simoneau, 2001; Macdermid, Fink & Stannard, 2015; Shultz, D'hondt, Fink, Lenoir & Hills, 2014). Customized non-linear integration method is also used (Alvarez, Alvarez, López & González, 2012).

The trapezoidal rule and Simpson's rule will be described as below.

#### Trapezoidal Rule

The Newton-Cotes formula is frequently used in numerical integration. In this formula, the function to be integrated is replaced by its approximation (e.g. a polynomial) which

can be integrated easier. The Newton-Cotes formula can be described as:

$$I = \int_a^b f(x)dx \cong \int_a^b f_n(x)dx \tag{2.1}$$

Where $f_n(x)$ is a polynomial and $n$ is the coefficient of the polynomial. It can be represented as:

$$f_n(x) = a_0 + a_1x + ... + a_{n-1}x^{n-1} + a_nx^n \tag{2.2}$$

The trapezoidal rule is known as the first-order Newton-Cotes formula., that is, $f_n(x) = f_1(x)$ in Equation 2.1, The trapezoidal rule can be generalized as:

$$I = \int_a^b f(x)dx \cong \int_a^b f_1(x)dx \tag{2.3}$$

The linear equation can be expressed as:

$$f_1(x) = f(a) + \frac{f(b) - f(a)}{b - a}(x - a) \tag{2.4}$$

Using the area under this line as an estimation of the integral:

$$I = \int_a^b \left[ f(a) + \frac{f(b) - f(a)}{b - a}(x - a) \right] dx \tag{2.5}$$

The result of the integral is:

$$I = (b - a)\frac{f(b) - f(a)}{2} \tag{2.6}$$

The integral can be divided into a partition of the interval, $[x_i, x_{i+1}] \in [a, b]$, such that $a = x_0 < x_1 < ... < x_{N-1} < x_N = b$, $\Delta x_n = x_n - x_{n-1}$, then applying the trapezoidal

rule to each sub-interval summing the results:

$$\int_a^b f(x)dx \approx \sum_{n-1}^{N} \frac{f(x_{n-1}) + f(x_n)}{2} \Delta x_n \tag{2.7}$$

When the partition increases, a better accuracy of approximation can be achieved.

Anwary, Yu and Vassallo (2017) obtained the velocity and position through trapezoidal rule. The error was less than 1% in the dataset with the best performance. However, for some dataset, the error increased to more than 30%.

**Simpson's Rule**

The Simpson's rule corresponds to the three-point Newton-Cotes rule. The three points $a$, $b$ and $m = \frac{a+b}{2}$ are selected to fit the primitive function with a parabola or other types of simple functions . Considering the computation time and accuracy, a quadratic function is used in the Simpson's rule. In Simpson's rule, $f(x)$ is primitive function and $g(x)$ is the fit function.

$$g(x) = Ax^2 + Bx + C \tag{2.8}$$

Then, $f(a) = g(a)$, $f(b) = g(b)$, $f(m) = g(m)$.

$$\begin{aligned} \int_a^b f(x)dx &\approx \int_a^b (Ax^2 + Bx + C)dx \\ &= \frac{A}{3}(b^3 - a^3) + \frac{B}{2}(b^2 - a^2) + C(b - a) \end{aligned} \tag{2.9}$$

This formula can be simplified as:

$$\int_a^b f(x)dx \approx \frac{(b-a)}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right] \tag{2.10}$$

### 2.2.2 Step frequency and acceleration

According to previous researches, the stride length varies with walking frequency, speed, acceleration or personal feature (weight, age, height, leg length, etc) (Cho & Park, 2006; Kao, Chen & Lin, 2001; R. Chen, Pei & Chen, 2011). In several pedestrian navigation systems (PNS), the stride length is estimated base on the walking frequency. Gusenbauer, Isert and Krösche (2010) considered the step frequency as the important parameter to estimate the step length. They used a linear equation to describe the relationship between frequency and stride length.

$$l_s = a + bf_s + \omega \tag{2.11}$$

Where $l_s$ is the estimated stride length, $f_s$ is the step frequency, $a$ and $b$ are constants, $\omega$ is defined as Gaussian noise $N(0, \sigma_s)$, $\sigma_s$ is the of the Gaussian noise distribution which is defined as:

$$\sigma_s = 0.24 - 0.09 f_s \tag{2.12}$$

Moreover, Qian, Ma, Ying, Liu and Pei (2013); Shin et al. (2007) combined step frequency with acceleration in the calculation the stride length using the following equations:

$$L = \alpha f + \beta v + \gamma \tag{2.13}$$

Where $\alpha$, $\beta$ are weighted parameters, $\gamma$ is a constant, $f$ is the frequency and $v$ is the acceleration. $f$ and $v$ can be obtained as:

$$\begin{cases} f_k = \frac{1}{t_k - t_{k-1}} \\ v_k = \sum_{t=t_{k-1}}^{t_k} \frac{(a_t - \bar{a}_k)^2}{N} \end{cases} \tag{2.14}$$

Where $t_i$ is the detected time duration of the $i^{th}$ step; $a_t$, $\bar{a}_k$ and $N$ represent the

acceleration data at time $t$, the average acceleration and the number of outputs on one step respectively.

### 2.2.3 Received signal strength

With the development of the Wireless Local Area Networks (WLANs), the WLAN access points (AP) are equipped in many places such as school, airport, shopping mall and museum. For indoor localization, the useful data is the received signal strength (RSS) from each AP. These information are available because the devices can use these the signal source as beacons when roaming within the network of APs.

In 1988, Motely and Keenan proposed the propagation model to estimate the distance by the radio signal strength (Motley & Keenan, 1988). It can be given by

$$P_{received}(d) = P_{received}(d_0) - 10\alpha log\left(\frac{d}{d_0}\right) \tag{2.15}$$

where $P_{received}(d)$ is the signal strength received by the device at the reference distance $d$, $P_{received}(d_0)$ is the signal strength received at the known distance $d_0$ and $\alpha$ is a coefficient for modeling of radio propagation in the environment. This model was used in several experiments but obtained poor results. In 2002, Chen and Kobayashi proposed the following refinement of this model which takes the effects of walls in the signal into consideration (Y. Chen & Kobayashi, 2002): .

$$P_{received}(d) = P_{received}(d_0) - 10\alpha log\left(\frac{d}{d_0}\right) + \sum_{i=0}^{N_w} n_i w_i \tag{2.16}$$

where $N_w$ is the number of walls, $n_i$ is the number of walls having an attenuation of $w_i\ dB$. This improved model obtained a better result.

Furthermore, Bahl and Padmanabhan (2000) provided a different approach called radio signal strength fingerprinting, it obtained better performance. It consists of

two step: offline stage and online stage. In the offline stage, it firstly establishes a map between collected received signal strength and certain positions in the indoor environment. Secondly, in order to collect fingerprints at various locations and set up the database, some on-site measurements need to be performed in designated areas. The offline stage also called training or profiling. In online stage, the location of the device is estimated by using the database set up in the offline stage.

## 2.3 Heading angle determination

After the gait cycle and the stride length being estimated, they can be combined with the heading of the pedestrian to determine his trajectory. So heading determination is another important issues for indoor navigation. In IMU-based localization system, the three-axis magnetometer and the three-axis gyroscope are commonly been used for heading angle measurement. The comparison between these two sensors are shown in Table 2.2. It can be seen that their advantages and disadvantages are complementary each other.,The gyroscope can correct the magnetic disturbances while the magnetic compass can determine and compensate for the errors in the gyroscope readings (Kim et al., 2004). The turning angle between two continuous steps can be defined as

Table 2.2: The comparison of the compass and gyroscope

| | Advantage | disadvantage |
|---|---|---|
| Magnetic compass | absolute azimuth | unpredictable external disturbances |
| | long term stable accuracy | |
| Gyroscope | no external disturbances | relative azimuth drift |
| | short term accuracy | |

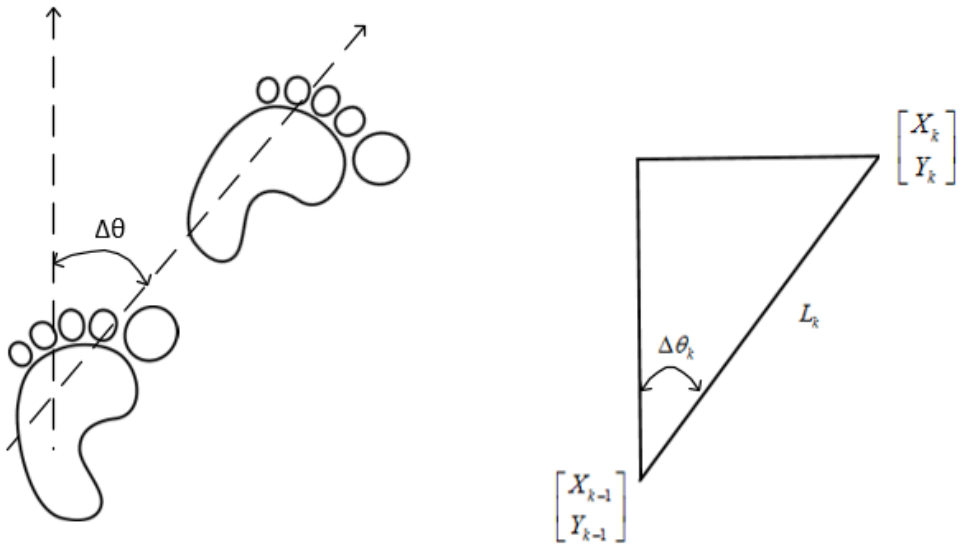



Figure 2.3: The illustration of the turning angle

where $x$ and $y$ are the position coordinates, $l$ is the stride length and $\theta$ is the heading angle. The differences between continuous steps at time $k$ and $k-1$ are denoted as $\Delta\theta_k = \theta_k - \theta_{k-1}$.

In the experiment of Liu, Dashti and Zhang (2013), the measurement error of magnetic compass as zero-mean Gaussian distributed $\delta_\theta \sim N(0, \sigma^2)$ are considered. Therefore, according to triangulation rule, the updated coordinates of the position can be calculated as

$$\begin{cases} x_k^l = x_{k-1}^l + L_k^l sin(\theta_k^l) \\ y_k^l = y_{k-1}^l + L_k^l cos(\theta_k^l) \end{cases} \tag{2.17a}$$

$$\begin{cases} L_k^l = L_{k-1}^l + \delta_L \\ \theta_k^l = \theta_{k-1}^l + \Delta_{\theta_k} + \delta_{\theta_k} \end{cases} \tag{2.17b}$$

Moreover, Li et al. (2012) also used a similar algorithm to calculate the heading angle: $\theta_i = \theta_i^e + \delta\theta_i$, where $\theta_i^e$ is the obtained direction of current step $i$ and $e^{th}$ is the

ground truth in $i$ step and $\delta$ is the noise of compass sensor readings. Particularly, for long distance estimation, they compared the changes in heading angle $\delta\theta_i$ with a given threshold $\theta_{th}$. They firstly calculated the mean heading angle of all steps since the previous turn, then compared the heading angle of current step with the mean heading angle .

$$\delta\theta_i = \begin{cases} \theta_i - \sum_{j=1}^{i-1} \theta_j, & |\delta\theta_i| \geq \theta_{th} \\ 0, & |\delta\theta_i| < \theta_{th} \end{cases} \tag{2.18}$$

It can be seen that if the change of heading angle is larger than the threshold, the turn at this step will be detected.

## 2.4 Summary

In this section, several previous works that related to our research are reviewed. Firstly, we reviewed the gait cycle detection algorithm. Secondly, the stride length estimation methods basing on different sources have been studied. Lastly, the heading determine method was reviewed. Most importantly, the previous researches provided us helpful guidance in different areas for our experiment.

# Chapter 3

# Theory

## 3.1 Introduction

In this chapter, theories related to stride length estimation will be reviewed. It includes an introduction of artificial neural network, several common signal filtering techniques to process IMU signals, coordinate systems used to describe the motion and the orientation.

## 3.2 Artificial Neural Network

Artificial neural network is a machine learning technology that simulates the neural network in a human brain and the associated intelligence. The neural network in the brain is a complex organization. There are approximately 100 billion neurons in an adult brain (Shatz, 1992). Typically, a neuron consists of nucleus, dendrites and axons (Pannese, 2015). A nucleus acts as a processing unit in the neuron. A dendrite acts as a postsynaptic component of the excitatory synapses in the brain. They played an essential role in synaptic transmission (Yuste & Denk, 1995). Moreover, an axon connects axon terminals and dendrites; those terminals are connected to other neurons to transmit information between the neurons. The neuron structure is shown in Figure 3.1 (Jain,

Mao & Mohiuddin, 1996).

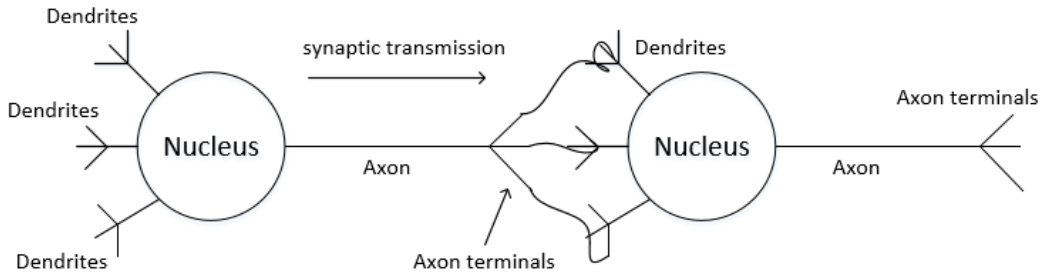



Figure 3.1: The illustration of neuron

### 3.2.1 McCulloch-Pitts neural model

After examining the structure of the neurons in human brains, McCulloch and Mathematician Pitts proposed a mathematical model , called McCulloch-Pitts (MP) model, of the neuron form in 1943 (McCulloch & Pitts, 1943). It contains three functions: input, output and process. They are analogous to dendrites, axon and nucleus respectively. In the input unit, each input parameter is assigned with a weight depends on the priority. The Figure 3.2 shows a typical neuron model: three inputs, one output and two calculation functions, where $a_i$ is the input and $w_i$ is the weight ( $i = 1,\ 2,\ 3$).

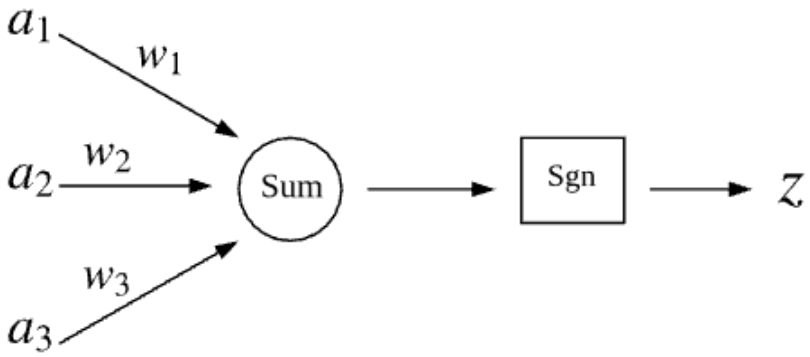



Figure 3.2: McCulloch-Pitts model of neural model

Sgn is a sign function which returns 1 if the output is a positive number, or returns 0

otherwise.

$$z = \mathrm{Sgn}(a_1 w_1 + a_2 w_2 + a_3 w_3) \tag{3.1}$$

Generally, the purpose of the neuron model is to predict the unknown attributes through the known properties (Gerstner & Naud, 2009). The known attributes are called feature, and the unknown properties to be predicted are called target. However, as the weights were set manually in advance, the MP model is a computational model other than a learning model (Buscema, 1998).

### 3.2.2 Perceptron neural network

In 1949, the neural network, named perceptron, consisting of two layers of neurons was proposed by Rosenblatt (Jain et al., 1996). The perceptron model was the first neural network which can be improved through the training process.

The perceptron model contains two layers: the input layer and the output layer, where the input unit is only responsible for data transmission and the output unit responsible for processing the information that is transmitted from input layer. The Figure 3.3 shows a perceptron neural network that contains two output units.

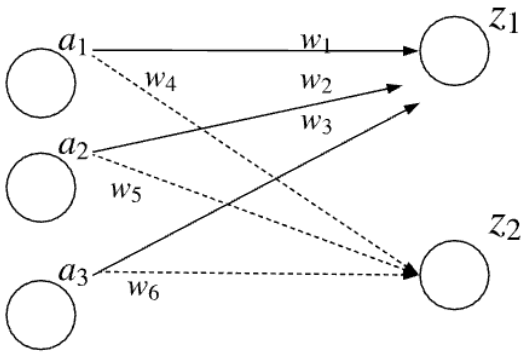



Figure 3.3: Perceptron neural network

Those weights are then associated with a weight matrix. The improved structure is shown in Figure 3.4.

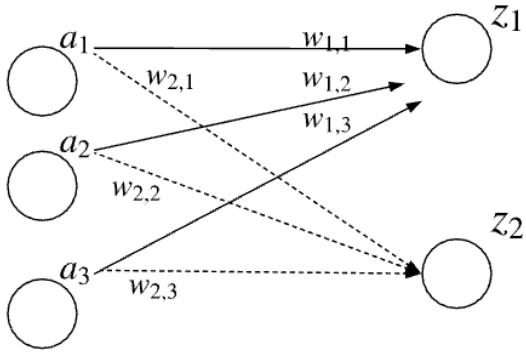



Figure 3.4: Improved perceptron neural network

$$z_1 = f(a_1 w_{1,1} + a_2 w_{1,2} + a_3 w_{1,3}) \tag{3.2a}$$

$$z_2 = f(a_1 w_{2,1} + a_2 w_{2,2} + a_3 w_{2,3}) \tag{3.2b}$$

Let $\vec{a} = [a_1 \;\; a_2 \;\; a_3^T]$ and $\vec{z} = [z_1 \;\; z_2 \;\; z_3]^T$; Then, the above formula can be rewrite as:

$$f(W\vec{a}) = \vec{z} \tag{3.3}$$

Through a training process, the weight of perceptron can be updated . Therefore, it can be considered as a logistic regression model (Jain et al., 1996) that is only able to do the simple linear classification task, as shown in Figure 3.5.

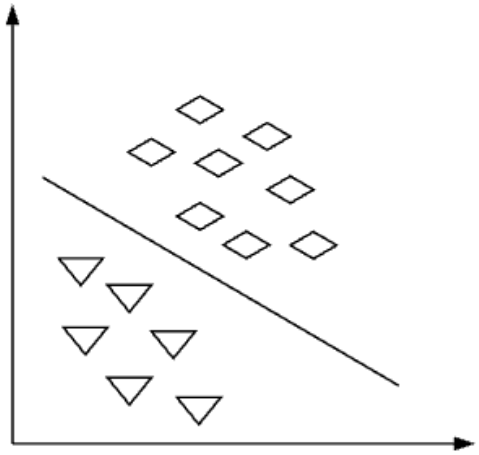


Figure 3.5: Linear classification

### 3.2.3 Two-layer neural network

The perceptron neural network cannot solve the problem of exclusive dis-junction (XOR). However, by using two layers neural network, this problem as well as nonlinear classification is resolved is solved. Rumelhar and Hinton and others proposed the backpropagation algorithm (BP) for training the neural networkJin, Li, Wei and Zhen (2000). BP neural network is a feed-forward neural network and is trained through error backpropagation algorithm. It has the ability to solve nonlinear approximation. The structure of two layers neural network contains an input layer (with an hidden layer), an output layer. As shown in Figure 3.6.

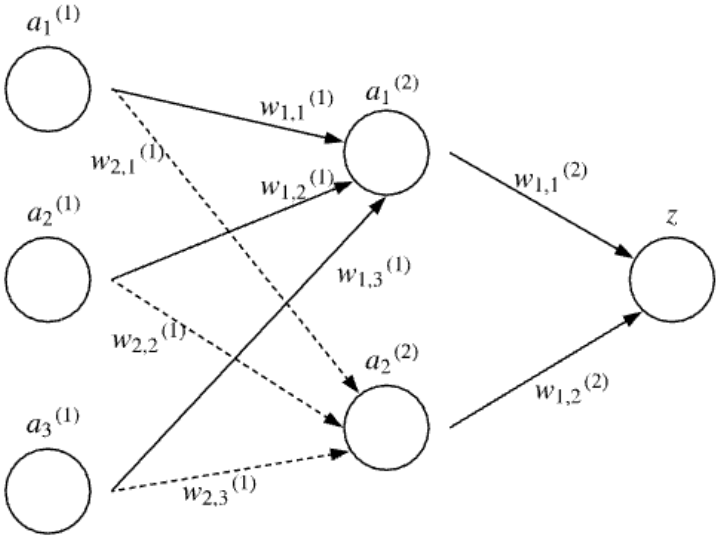



Figure 3.6: Two layers neural network

$a_i{}^{(j)}$ represents the output of the $i^{th}$ node of layer $j$. The nodes in second layer can be derived from the output of the first layer with the first weight matrix $w^{(1)}$ and function $f$. Where function $f$ is the activation function, that define the output of the node. And the output $z$ can be derived from the second weight matrix $w^{(2)}$ and the same activation function $f$. In the standard two-layer ANN, The activation function $f$ is a normalizable sigmoid function. This function normalized the output into a range of $(0, 1)$ (Jain et al., 1996).

$$a_1{}^{(2)} = f\left(a_1{}^{(1)}w_{1,1}{}^{(1)} + a_2{}^{(1)}w_{1,2}{}^{(1)} + a_3{}^{(1)}w_{1,3}{}^{(1)}\right) \tag{3.4a}$$

$$a_2{}^{(2)} = f\left(a_1{}^{(1)}w_{2,1}{}^{(1)} + a_2{}^{(1)}w_{2,2}{}^{(1)} + a_3{}^{(1)}w_{2,3}{}^{(1)}\right) \tag{3.4b}$$

$$z = f\left(a_1{}^{(2)}w_{1,1}{}^{(2)} + a_2{}^{(2)}w_{1,2}{}^{(2)}\right) \tag{3.4c}$$

When adding node to output layer, we use matrix and vector to express the variable quantity. $a^{(1)}$, $a^{(2)}$ and $\vec{z}$ are vector data that is transmitted in the network; $w^{(1)}$ and $w^{(2)}$ are weights matrix parameters .

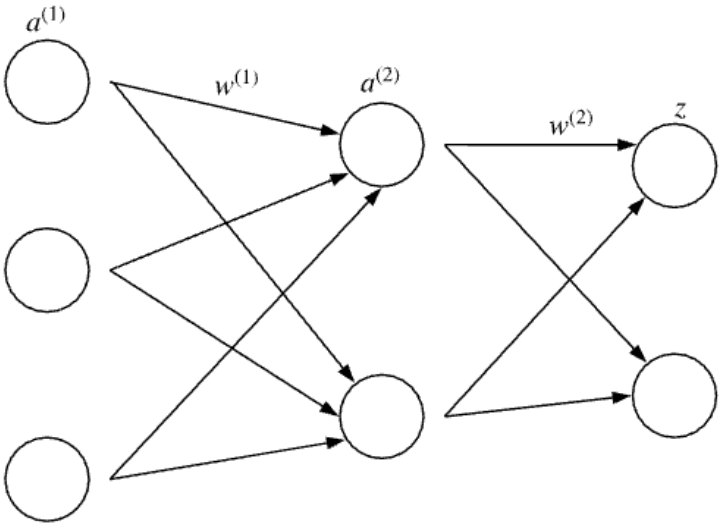



Figure 3.7: Improved two layers neural network

It can be expressed as:

$$f(\overrightarrow{w^{(1)}} * \overrightarrow{a^{(1)}}) = \overrightarrow{a^{(2)}} \tag{3.5a}$$

$$f(\overrightarrow{w^{(2)}} * \overrightarrow{a^{(2)}}) = \vec{z} \tag{3.5b}$$

A two layers neural network is able to approximate any continuous function. The original data is spatial transformed by the hidden layer, especially the parameter matrix of the hidden layer, then the data can be classified by linear classification. In essence, the two-layer neural network simulates the nonlinear function by two-layer linear model.

### 3.2.4 Multi-layer neural network

The multi-layer neural network allows multiple sub-layers in the hidden layer (Svozil, Kvasnicka & Pospichal, 1997). It. In particular, not only the nodes, but also the sub-layer number in hidden layer can be changed. As shown in Figure 3.8, there are three weight matrices in this network, six in $w^{(1)}$, four in $w^{(2)}$ and six in $w^{(3)}$.

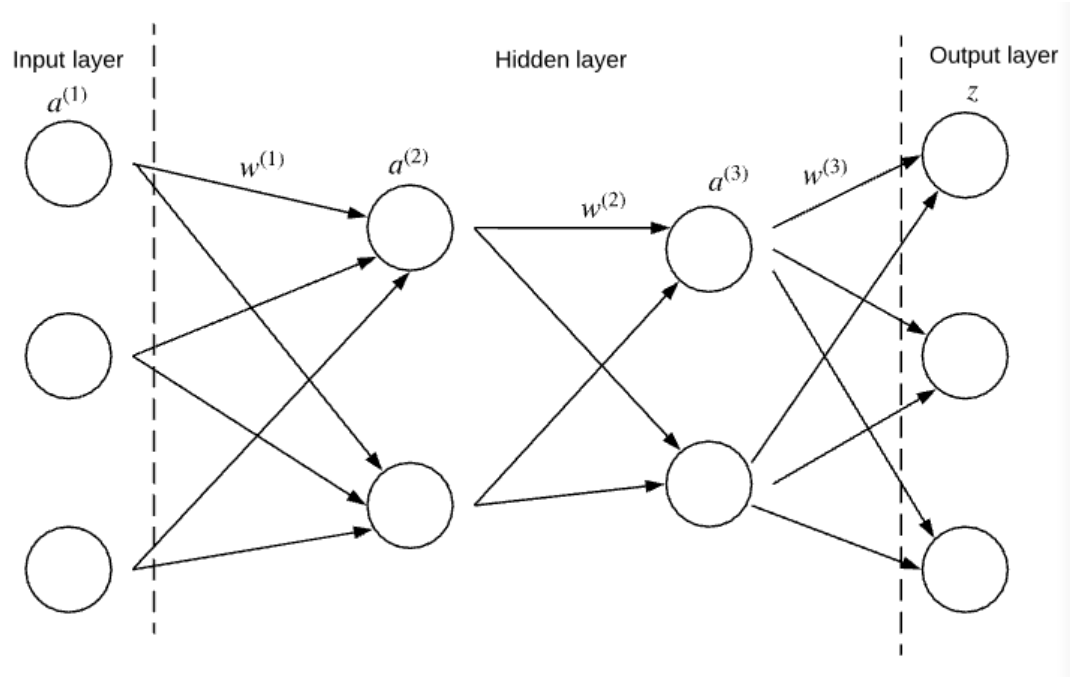



Figure 3.8: Multilayer neural network

As shown in Figure 3.9, we can increase the node number in the hidden layer. Murata, Yoshizawa and Amari (1994) found that an important but challenging problem is to determine the suitable number of hidden units by using only input and output samples. According to Tetko, Livingstone and Luik (1995), with more neurons in the hidden layer, if the input samples for the ANN are included in the training data set, the errors between the output value and the true value will be reduced. On the other hand, if input samples for the ANN are not included in the training data set, the errors between the output value and the true value will increase. This is called "over-fitting". Supposing we have two networks, as described in (Figure 3.8 and Figure 3.9) with same numbers of hidden layer but different neuron number, if the input data set that included in the training dataset, the network with fewer neuron number may have worse performance than the network with more neuron number after enough training. However, if the input dataset is not included in the training dataset, the network with fewer neuron number may have better performance.

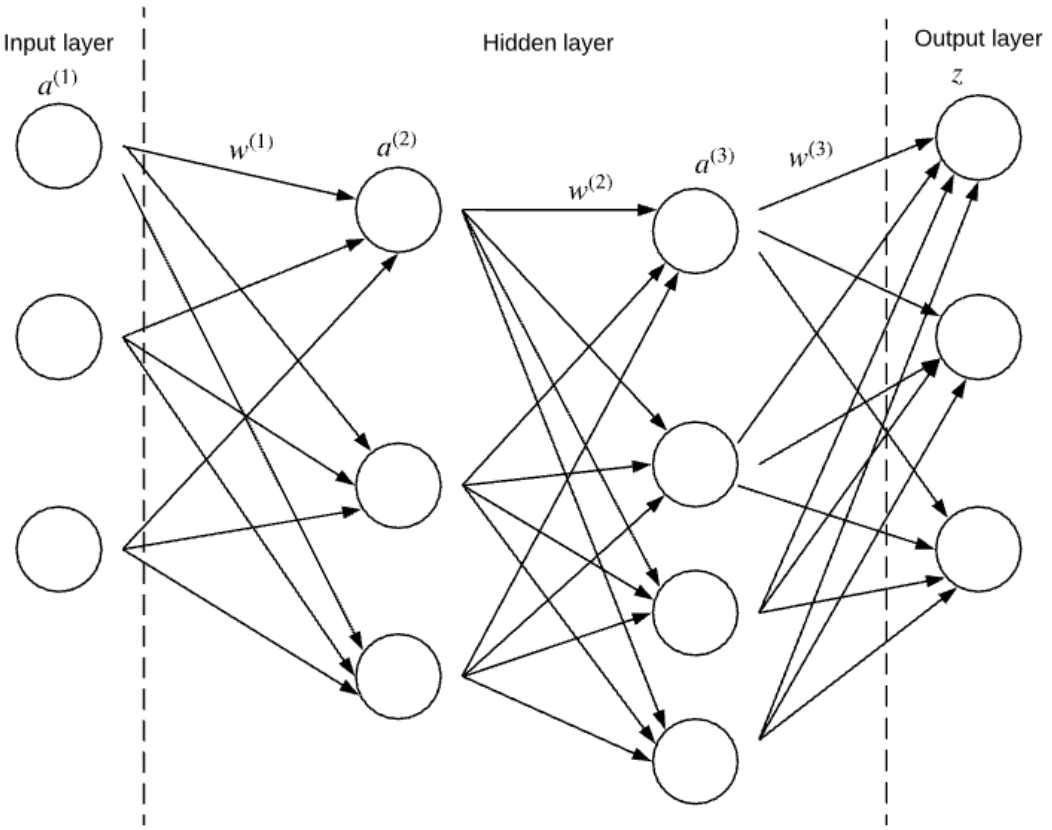



Figure 3.9: Multilayer neural network(neuron number increased)

As shown in Figure 3.10 where the number of hidden layers in the neural network is twice what is in Figure 3.10, , it can be more accurate to extract the features and has better ability to fit the correlation between the true value and actual output (Panchal, Ganatra, Kosta & Panchal, 2011).

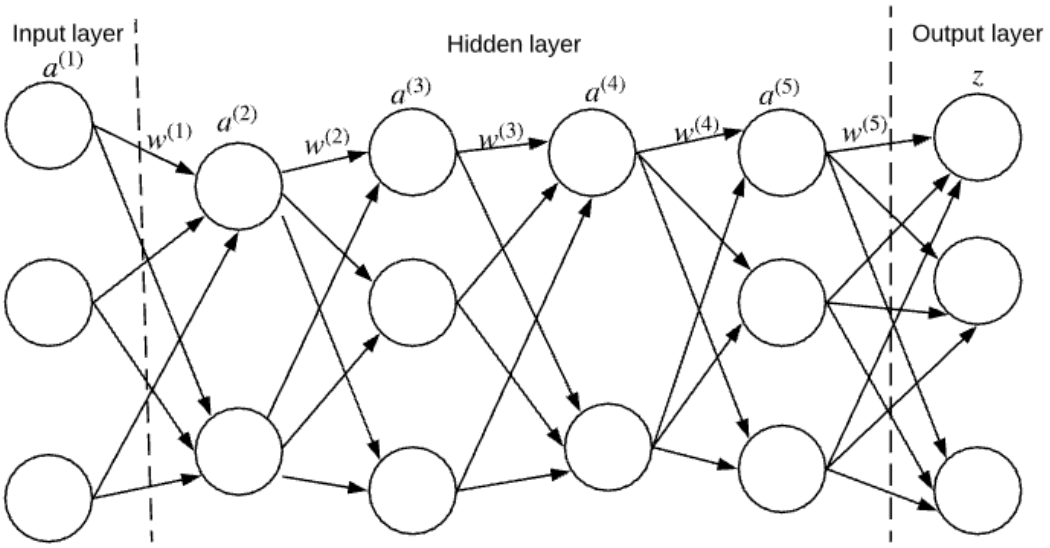



Figure 3.10: Multilayer neural network(layer number increased)

### 3.2.5 ANN MATLAB toolbox

The MATLAB (version 2017b) ANN toolbox is used to perform the ANN classification and fitting process in the experiment. In the toolbox, there are several training algorithms for the multilayer neural network can be selected for different tasks such as fitting and classification problems.

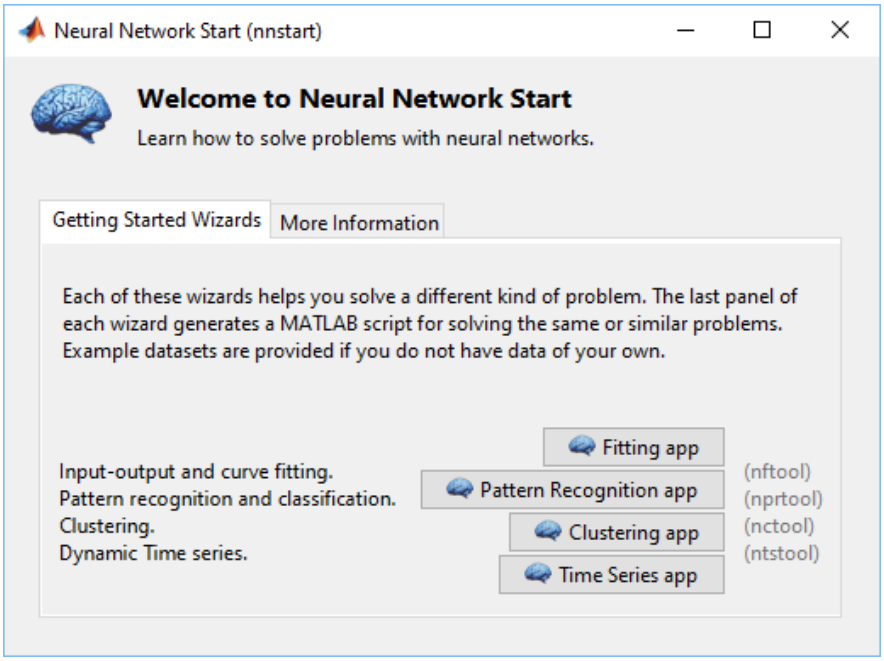



Figure 3.11: ANN toolbox in MATLAB (version 2017b)

For ANN fitting, there are three algorithms namely, Levenberg-Marquardt backpropagation Bayesian regularization backpropagation and Scaled conjugate gradient backpropagation training algorithm. The Levenberg-Marquardt backpropagation training algorithm uses the Levenberg-Marquardt optimization to update the weight and bias of input parameters. It calculate the Jacobian matrix of the weight and bias of the variable and adjust the value according to Levenberg-Marquardt optimization (Roweis, 1996). Bayesian regularization backpropagation training function also uses Levenberg-Marquardt optimization (Roweis, 1996) to adjust the weight and bias of the input variable in ANN. Moreover, it uses the Bayesian regularization (Aggarwal, Singh, Ch & Puri, 2005) in the end of the ANN training to provide a generalized qualities of the trained network. Instead of using Levenberg-Marquardt optimization to update the weight and bias, Scaled conjugate gradient backpropagation training algorithm uses the scaled conjugate gradient method (Møller, 1993) to update the weight and bias of the input variables.

For ANN classification, there are three algorithms can be used in the MATLAB ANN toolbox. Similar to the ANN fitting, Levenberg-Marquardt backpropagation and Scaled conjugate gradient backpropagation training algorithm also can be used in ANN classification. Instead of fitting the value, the ANN classification function will use the fitting value to distinguish the Classes in the input data. ANN classification function also uses the gradient to update the weight and bias of the variable in the input layer. Different from ANN fitting, Gradient descent backpropagation can be used for ANN classification. It uses gradient descent with adaptive learning rate to update the weight and bias of the input variable in ANN (Hagan, Demuth, Beale & De Jesús, 1996).

Overall, in ANN fitting and classification, there are four different training algorithm can be used. Experiments will be conducted to test the performance and determine the optimal algorithms for estimating the stride length.

## 3.3 Noise Filtering Method and Smoothing Method

The acceleration and Euler angles signals from the IMU usually contain a small offset even when the sensor is stationary. This is known as the sensor bias (Cruz, Alouani, Rice & Blair, 1992). When the sensor attached to the object, the noise of the output signal is not only influenced by sensor bias, but also the movement of the object. In order to remove the noise of the output signal from the IMU, noise filtering and signal smoothing method need to be applied before the data are processed

### 3.3.1 Moving average filter

The moving average filter (MA filter) generates the output signal by averaging multiple input signals (Sato, 2001). It can be expressed as:

$$y(i) = \frac{1}{M} \sum_{j=0}^{M-1} x(i+j) \tag{3.6}$$

where $y$ is the output signal, $x$ is the input signal and $M$ is the number of averaged input points. For example, we take previous three input points and the current point to calculate the average value of these four points.

$$y(n) = \frac{1}{4}x(n) + \frac{1}{4}x(n-1) + \frac{1}{4}x(n-2) + \frac{1}{4}x(n-3) \tag{3.7}$$

Alternatively, the input points can be symmetrically selected around the current point. Moving average filter is a common tool for filtering data. Due to its simplicity, MA filter is optimal for reducing random noise. However, it ignores the relationships between these points, which may cause data distortion (Azami, Mohammadi & Bozorgtabar, 2012).

### 3.3.2 Savitzky-Golay Filtering

The Savitzky-Golay (S-G) filter, published in 1964, is a specialized digital low-pass filter also called S-G smoother based on local least-squares polynomial approximation (Schafer, 2011)Savitzky and Golay (1964). In order to avoid the data distortion, S-G filter uses the linear least square to continuously fitting a polynomial to a subset of selected data. This process is called "convolution". Also, Savitzky and Golay (1964) introduced the "convolution coefficients table" to pair with different polynomial types and subset size.

The S-G filter can be expressed as:

$$Y_j = \sum_{i=-\frac{m-1}{2}}^{\frac{m-1}{2}} C_i y_{j+i}, \quad \frac{m-1}{2} \leq j \leq n - \frac{m-1}{2} \tag{3.8}$$

The data consists of a set of $n(x_j, y_j)$ points where $j = 1, 2, ..., n$ and $x$ is an independent variable, $y_j$ is an observed value and $m$ is the subset size. The convolution coefficients $C_i$ can be checked in the convolution coefficient table (Table 3.1) that formulated by Savitzky and Golay (1964).

Table 3.1: Convolution coefficient

| Polynomial Degree | quadratic or cubic 2 or 3 | | | quartic or quintic 4 or 5 | |
|---|---|---|---|---|---|
| Window size | 5 | 7 | 9 | 7 | 9 |
| -4 | | | -21 | | 15 |
| -3 | | -2 | 14 | 5 | -55 |
| -2 | -3 | 3 | 39 | -30 | 30 |
| -1 | 12 | 6 | 54 | 75 | 135 |
| 0 | 17 | 7 | 59 | 131 | 179 |
| 1 | 12 | 6 | 54 | 75 | 135 |
| 2 | -3 | 3 | 39 | -30 | 30 |
| 3 | | -2 | 14 | 5 | -55 |
| 4 | | | -21 | | 15 |
| Normalisation | 35 | 21 | 231 | 231 | 429 |

For example, when using a seven-points quartic polynomial for smoothing, $m = 7$, $i = -3, -2, -1, 0, 1, 2, 3$ and the $j^{th}$ smoothed data point $Y_j$ is given by

$$Y_j = \frac{1}{231}(5y_{j-3} - 30y_{j-2} + 75y_{j-1} + 131y_j + 75y_{j+1} - 30y_{j+2} + 5y_{j+3}) \tag{3.9}$$

where $C_{-3} = 5/231$ and $C_2 = -30/231$, etc.

The principle of the S-G filter can be summarized in three steps:

1. Finding a suitable least-square fit for each subset of the signal.

2. Using the coefficient of the polynomial replaces each data point.

3. Computing numerical derivatives from each fitted polynomial at each data point.

### 3.3.3 Local weight regression

The local weight regression (LWR) is a method for smoothing by fitting curves and surfaces of the data (Cleveland & Loader, 1996). It models the regression functions of independent and dependent variables without any functional relationship between the previously specified variables. To use local regression smoothing, we compute the regression weight $w_j$ of each data point. The predictive value $x$ is associated with the response value that needs to be smoothed, $x_j$ are proximal points of $x$ within the span, $d(j)$ is the span or bandwidth. Then, the LWR filter output is obtained by continuously fitting a polynomial to the weighted linear least-squares regression.

$$w_j = \left(1 - \left|\frac{x - x_j}{d(j)}\right|^3\right)^3 \tag{3.10}$$

Due to the nature of the least square, the disadvantage of LWR is that it can be strongly affected by outliers. Since local regression typically involves a subset of a complete data set, the problem of outliers is exacerbated. To solve this problem, we use robust local weighted regression which includes an additional calculation of robust weights to eliminate outliers and smooth the data According to Nurunnabi, West and Belton (2016). The robust procedure is consisted by the following steps:

1. Calculating the residuals by using the robust procedure.

2. Computing the weights of each point.

$$w_i = \begin{cases} \left(1 - \left(\frac{r_i}{6MAD}\right)^2\right)^2, & |ri| < 6MAD \\ 0, & |ri| \geq 6MAD \end{cases} \tag{3.11}$$

where $r_i$ is the residual of data at time $i$ which is computed at the first step.

$$MAD = median(|r|) \tag{3.12}$$

$MAD$ is absolute deviation of residual. It can reflect the discreteness of dataset. We compare $r_i$ to $6MAD$: the weight of the robust is approaching to 1 if $r_i$ is smaller than $6MAD$. In the contrary, if $r_i$ is larger than $6MAD$, the associated point is excluded and the robust weight is 0. In this way, the effects of the outliers can be reduced.

3. Using the robust weight that is calculated in the previous steps and local regression weight to smooth the data.

4. Repeating the previous two steps until the estimated coefficient value is converged.

According to the experiment conducted by Cleveland (1979), by repeating these two steps twice, the effects of outliers on the fitting process can be reduced.

## 3.4 Orientation and coordination

### 3.4.1 Orientation of INS

#### Euler angles

The Euler angles are the three angles introduced by Euler (Weisstein, 2009) to describe the rotation of the rigid body. Formally, it is a three-dimensional vector whose values represent the three-axis rotation angle of the object respectively. Specifically, the Euler angles follow the $Z - Y - X$ sequence. These three angles are known as roll, pitch and yaw. Therefore, the sequence of the rotations represents first the yaw angle, followed by pitch and roll angle.

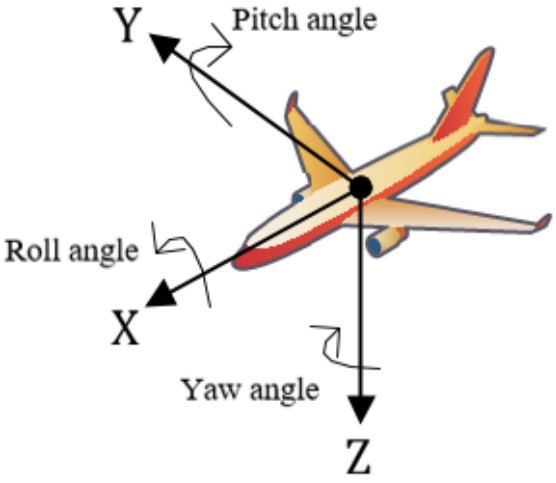



Figure 3.12: Euler angles

These three angles and the sequences are defined below:

- The yaw denoted as $\psi$,represents a rotation around Z-axis. After rotating around the Z-axis, the new frame axes are called $X_1, Y_1, Z_1$.
- The pitch angle denoted as $\theta$ represents a rotation around $Y_1$. With yaw and pitch rotation, the new frame axes are called $X_2, Y_2, Z_2$.
- The roll angle denoted by $\phi$, represents a rotation around $X_2$. With yaw, pitch and roll rotations, the new frame axes are named $X_3, Y_3, Z_3$.

The graphical illustration is:

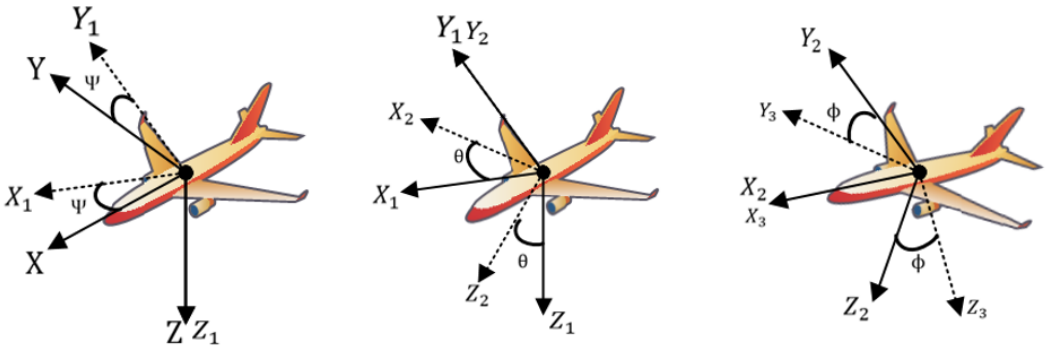



Figure 3.13: Yaw-pitch-roll rotation sequence

The relative rotation matrix are given as follow:

$$R_1(\psi) = \begin{pmatrix} cos(\psi) & sin(\psi) & 0 \\ -sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.13a}$$

$$R_2(\theta) = \begin{pmatrix} cos(\theta) & 0 & -sin(\theta) \\ 0 & 1 & 0 \\ sin(\theta) & 0 & cos(\theta) \end{pmatrix} \tag{3.13b}$$

$$R_3(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & sin(\phi) \\ 0 & -sin(\phi) & cos(\phi) \end{pmatrix} \tag{3.13c}$$

However, Eluler angles representation has a similarity which is also called gimbal lock. The gimbal lock occurs when one of the axes in fixed coordinate axes are concentric with the axes in rotated coordinate. Therefore, Hamilton firstly proposed quaternions in 1843 to deal with this problem Hamilton (1866).

### Quaternions

The Quaternions are a number system that extends the complex numbers. It can be generally represented as $q = a + bi + cj + dk$ where $i, j, k$ are the imaginary units, $i^2+j^2+k^2 = ijk = -1$ and $a, b, c, d$ are real numbers. Generally, $a$ is called the scalar part and three components $b, c, d$ taken together as a three-dimensional vector is called the vector part of a quaternion. Thus quaternions can be written as $q = (a, (x, y, z)) = (a, \vec{u})$. The figure below shows that the point $P$ is rotated around the point $Q$ then obtain a new point $P'$, the rotated angle denoted as $\theta$. The rotation around point $Q$ can be represented as a rotation around the axis $\vec{u}$, which is passing through that point. Compared with Euler angles, the quaternions are easier to calculate . By using the Euler angles, all three rotations must be measured, whereas the quaternions only require to measure one angle of rotation around an axis. (Hamilton, 1866).

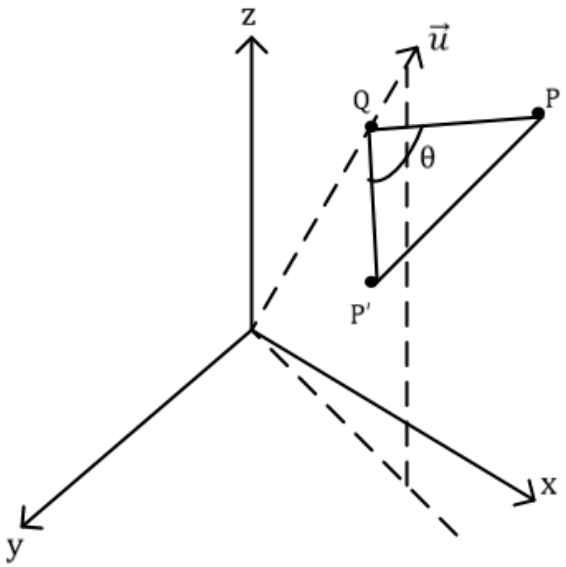



Figure 3.14: Rotation around specified axis

where $\vec{u} = (u_x, y_y, u_z)$ and $\vec{u}$ is an unit vector, $q = (\vec{u} * sin\frac{\theta}{2}, cos\frac{\theta}{2})$. Then we extended the point $P$ to quaternion space, $p = (0, x, y, z)$. The conjugate complex number of $q$ is $q' = (-\vec{u} * sin\frac{\theta}{2}, cos\frac{\theta}{2})$. Finally, we can calculate the coordination of new

point $P'$ by using the formula $P' = qpq'$

### 3.4.2 Coordinate systems

Several coordinate systems are widely used to represent the orientation and the position of the IMU such as the earth-centered earth-fixed (ECEF) coordinate system, the local north-east-down (NED) coordinate system, vehicle-carried NED frames and body frame.

#### Earth-centered earth-fixed coordinate system

The ECEF coordinate system also called earth-centered rotational coordinate system (Cai, Chen & Lee, 2011). It has been widely used, for example, it is the primary coordinate system of GPS. The origin and axes of this frame are shown in Figure 3.15.

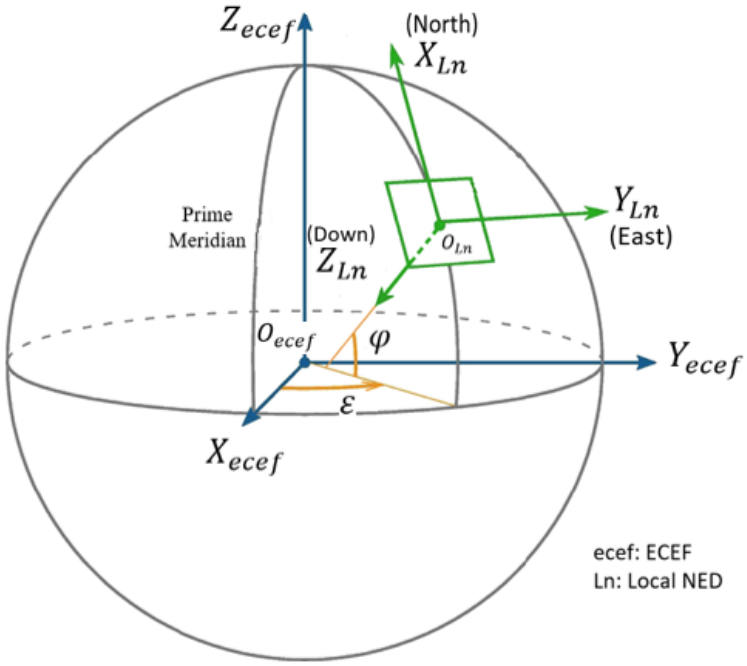



Figure 3.15: Earth-centered earth-fixed and local north-east-down coordinate systems

where $O_{ecef}$ is the origin point $(0, 0, 0)$, it is defined as the center of the earth. The x-axis ($X_{ecef}$) intersects the sphere of the equator and prime meridian, z-axis ($Z_{ecef}$)

extends through the north pole and y-axis ($Y_{ecef}$) is orthogonal to the z-axis and the x-axis. This frame rotates with the earth, thus it is convenient for positioning geostationary objects.

### Local north-east-down (NED) coordinate system

The local NED coordinate system is also known as the ground coordinate system (Cai et al., 2011). The origin and axes are shown at Figure 3.15. In this coordinate system, we assume the earth is flat. It can be seen that this coordinate system is fixed to the surface of the earth. Its origin is usually chosen to be the center of gravity of the device. The x-axis ($X_{Ln}$) represents the position along the geodetic north. The y-axis ($Y_{Ln}$) points toward the geodetic east and z-axis ($Z_{Ln}$) represents vertical position. This coordinate system is usually used for small areas where the curvature of the earth is not considered.

### Vehicle-carried NED coordinate system

The vehicle-carried NED coordinate system is associated with the devices such as sensor, phone and vehicle (Cai et al., 2011). As shown in Figure 3.16, the origin ($O_{nv}$) is located at the center of gravity of the device. The x-axis ($X_{vn}$) represents the geodetic north, the y-axis ($Y_{vn}$) points toward the geodetic east and the z-axis ($Z_{vn}$) represents vertical position.

The axial direction of the vehicle-carried NED coordinate system changes with respect to the movement of the device, and therefore does not coincide with the axial direction in the local NED frame. However, many sensor-based devices are mounted on small vehicles such as micro-rotorcraft, they only operate in a small region with a low speed and the direction differences is completely ignored. Therefore, in this case, these two coordinate systems are consistent with each other.

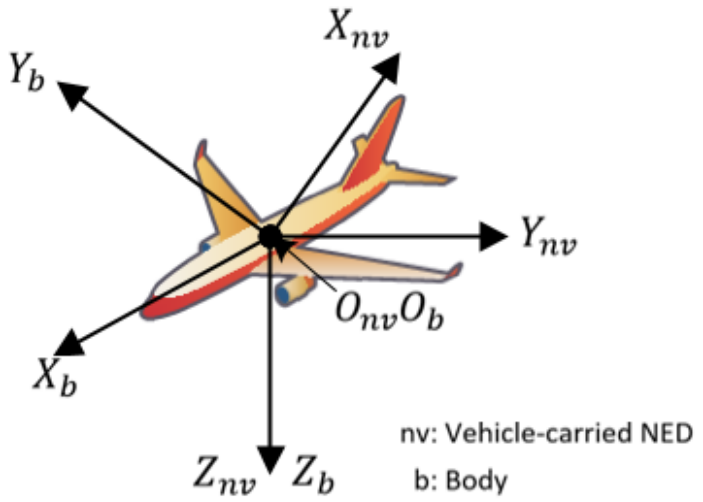



Figure 3.16: Body coordinate system and vehicle-carried NED coordinate system

**Body coordinate system**

The body frame is directly defined by the body of the devices (Cai et al., 2011). As shown in Figure 3.16, this system uses the right-hand rule: the origin ($O_b$) is at the same location as ($O_{nv}$), the center of gravity of the device, the x-axis ($X_b$) is consistent with the direction of the device, y-axis ($Y_b$) points forward the right side of the device and z-axis ($Z_b$) is orthogonal to the x-axis and y-axis.

## 3.5 Summary

In this section, several related theoretical knowledge relevant to our research are reviewed. We firstly studied artificial neural network, followed by the noise filtering algorithm. Then, different coordinate systems are also reviewed.

# Chapter 4

# Data collection

## 4.1 Equipment

The main equipment used in our experiments is the Next Generation IMU (NGIMU). It equips on-board sensors such as gyroscope, accelerometer and magnetometer. All measurements in NGIMU are time-stamped and all sensors are pre-calibrated in factory. The measurements of orientation such as Euler angles, rotation matrix and quaternion are presented in the vehicle-carried NED coordinate frame. It also provided the linear acceleration with gravity removed. This device can be accessed by USB and Wi-Fi. In addition, the data can also be logged to an on-board SD card (x-io Technologies Ltd, 2017). The NGIMU uses Open Sound Control (OSC) protocol to communicate with the other equipment, it also provides libraries for a number of programming languages, such as JAVA, C++ and Python (x-io Technologies Ltd, 2017).

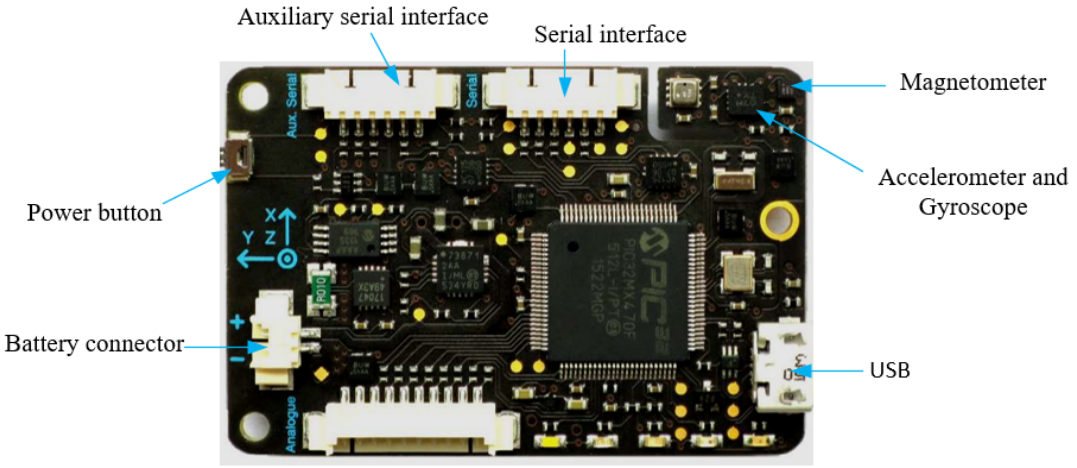



Figure 4.1: The layout of NGIMU

The NGIMU uses a right-handed coordinate system as it is shown in Figure 4.2 (x-io Technologies Ltd, 2017).

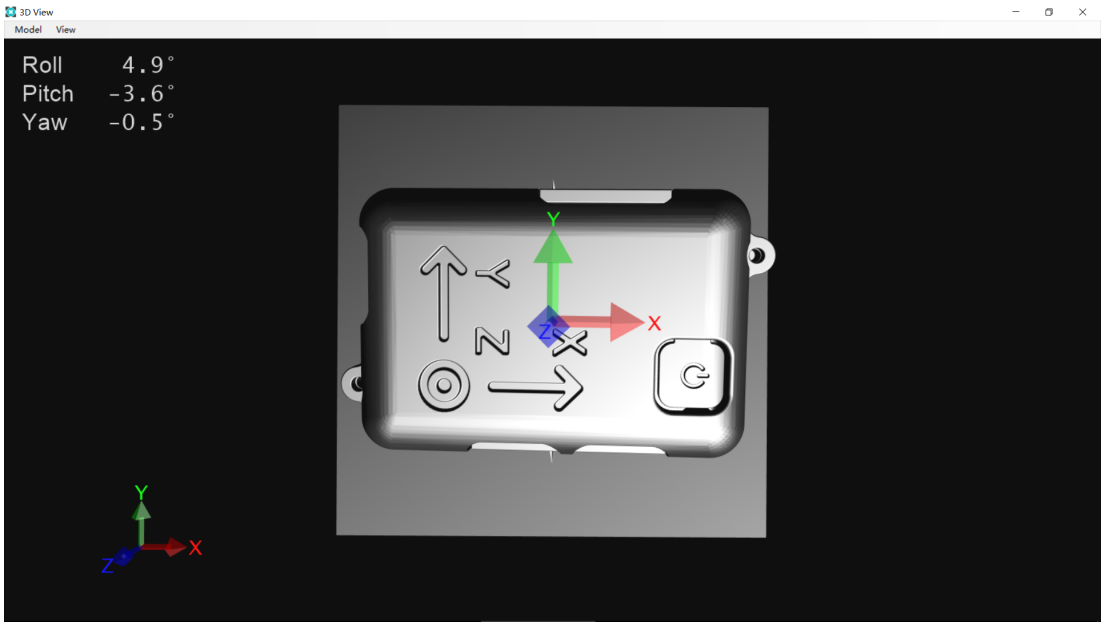



Figure 4.2: IMU coordinate system

## 4.2 Communication

In our experiments, a Wi-Fi access point (802.11n, 5 GHz, AP mode) is used to transmit the data from the sensor to the computer in real-time. Data such as acceleration and Euler angle are collected different OSC addresses. The example of OSC address interpretation is shown in Table 4.1.

Table 4.1: OSC address interpretation

| OSC address | Message |
|---|---|
| /sensors | Gyroscope, accelerometer, magnetometer |
| /quaternion | The quaternion |
| /matrix | The rotation matrix |
| /euler | The Euler angles |
| /linear | The linera acceleration |
| earth | The earth acceleration |

Furthermore, each OSC address contains different arguments. For example, the OSC address $/quaternion1$ contains the first argument which is the $a$ element of the quaternions. An example of the interpretation of arguments are shown in Table 4.2.

Table 4.2: Interpretation of the quaternion message arguments

| Argument | Type | Description |
|---|---|---|
| 1 | float32 | Quaternion a element |
| 2 | float32 | Quaternion x element |
| 3 | float32 | Quaternion y element |
| 4 | float32 | Quaternion z element |

The sampling rates of the gyroscope and accelerometer in NGIMU are fixed at 400 Hz whereas magnetometer is fixed at 100 Hz. However, the data sent rate of the IMU can be specified by the user. In this project, in order to balance performance and stability, we set the it to be 40Hz. All data from the IMU are sent as a time-stamped OSC bundle containing a single OSC message. As shown in Figure 4.3, we can use the command $/rate/OSC\ address, send\ rate$ to reset the send rate (x-io Technologies Ltd, 2017).

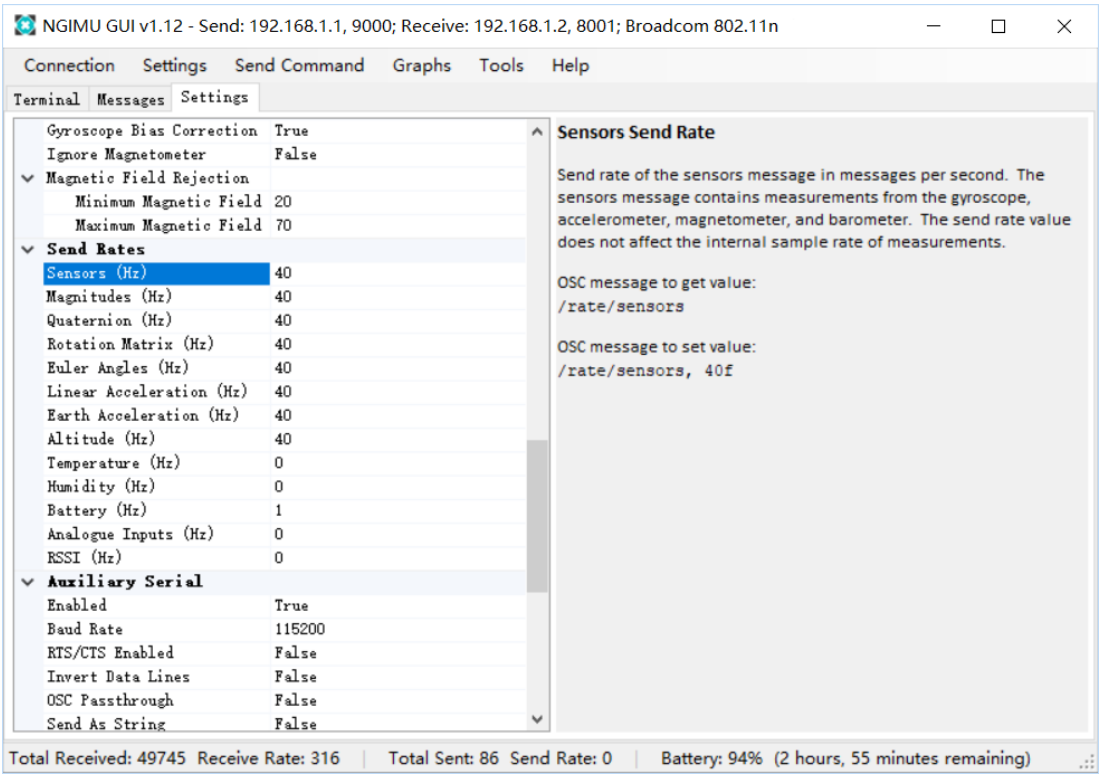



Figure 4.3: Send rates setting

## 4.3 ANN classification training dataset

First of all the test field is set up. As shown in Figure 4.4, there are several 5cm ×7.5cm rectangle stickers were attached to the floor. There are three groups of stickers representing three classes of data sets respectively. From right to left are Class One, Class Two and Class Three, representing $120 \pm 5cm$, $130 \pm 5cm$ and $140 \pm 5cm$ respectively. Therefore, the maximum error among each class is $\pm 5cm$. As shown in Figure 4.5, the IMU is attached to the heel of boot of the person who does the test. When collecting the data, the heel step must be on the sticker . Because the stride length is the distance between the constant initial contact points of the same foot, and the sticker width is 5cm, when heel step on the sticker, the maximum gap between minimum step length and maximum step length is 10cm. Therefore, the maximum error of our data collection

for ANN classification training is 10cm.

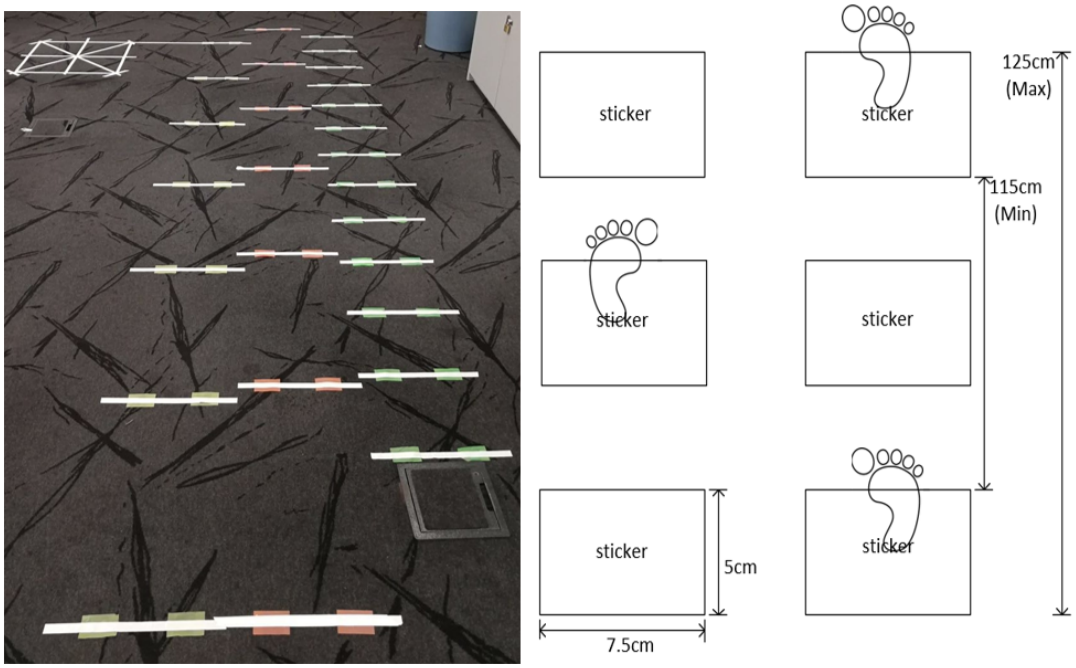



Figure 4.4: Test field (left); Step length measurement (right)

Figure 4.5: IMU attached to boot

The data for every three steps are stored in one Excel file. As the IMU is running at

40Hz and the testing time is 8 seconds, each file contains approximately 320 data points at an interval of 0.025s. Each datum contains time-stamp, acceleration(body frame), Euler angles, quaternions, rotation matrix, earth acceleration and linear accelerations. Shown in Table 4.3.

Table 4.3: Interpretation of ANN classification data

| | | | |
|---|---|---|---|
| Column 1 | Time-stamp | Column 9 | Quaternion x element |
| Column 2 | Acceleration in the sensor x axis | Column 10 | Quaternion y element |
| Column 3 | Acceleration in the sensor y axis | Column 11 | Quaternion z element |
| Column 4 | Acceleration in the sensor z axis | Column 12-20 | Rotation matrix in row-major order |
| Column 5 | Roll (x) angle in degrees | | |
| Column 6 | Pitch (y) angle in degrees | Column 21 | Acceleration in the Earth x axis |
| Column 7 | Yaw/heading (z) angle in degrees | Column 22 | Acceleration in the Earth y axis |
| Column 8 | Quaternion w element | Column 23 | Acceleration in the Earth z axis |

The stride length is considered to be related to linear acceleration and Euler angles in the movements. Linear acceleration is the gravity-free acceleration in vehicle-carried NED coordinate frame (sensor coordinate frame). At the beginning, goose step is used as it is assumed to generate less noises in the sensor readings. The linear acceleration data of walking in goose step, which the knees are not bent while walking, is shown in Figure 4.6. It is shown that the goose step actually cannot reduce the noises as the data contain a number of large spikes in each step. Alternatively the walking in the normal step, which the knees are bent while walking and the foot landed on the ground softly, is tested . The linear acceleration in the normal step is shown in Figure 4.7 where there is not significant spikes. This is due to the fact that in the normal steps, the foot lands the floor softly.

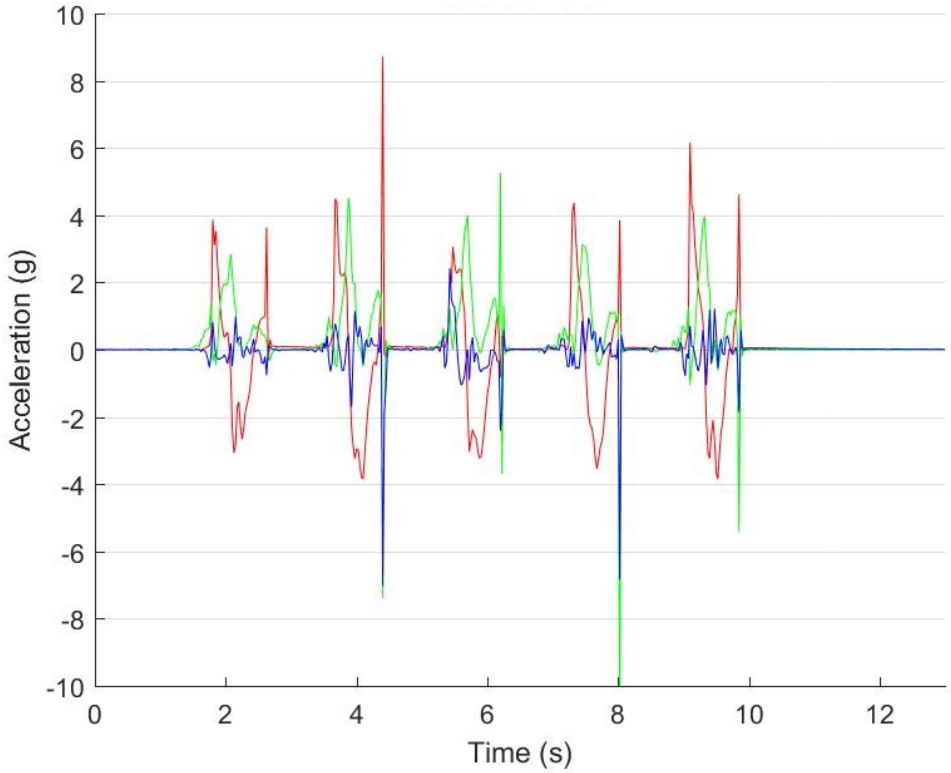



Figure 4.6: Three-axis linear acceleration of goose step

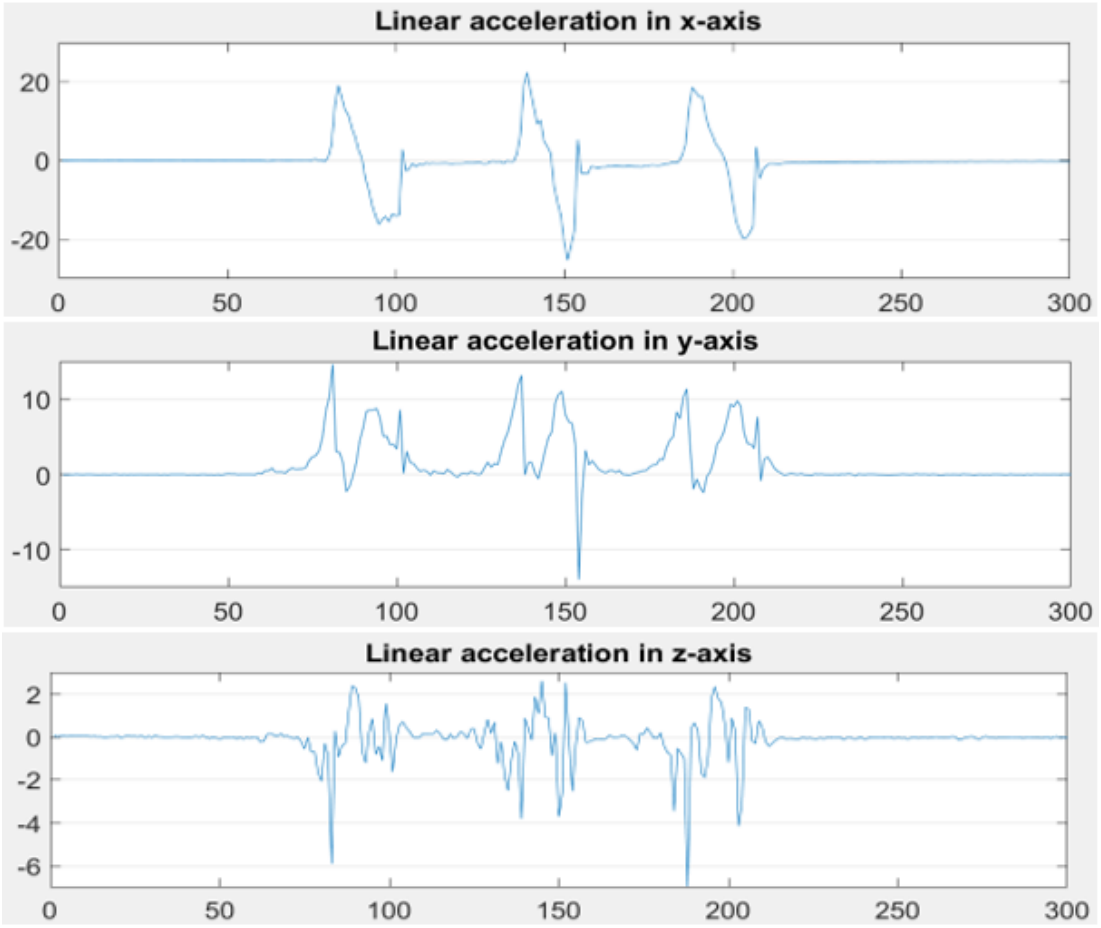



Figure 4.7: Three-axis linear acceleration of normal gait

From Figure 4.7 and Figure 4.8, we can clearly distinguish the first step, the second step and the third step via the acceleration data. The three-axis linear acceleration shows that the x-axis acceleration (horizontal acceleration ) and y-axis acceleration (vertical acceleration) change regularly while the z-axis acceleration changes irregularly. When one walks along a straight line on the ground, the x-axis acceleration has a significant influence to the stride length. Therefore, it is a main element in the data processing.

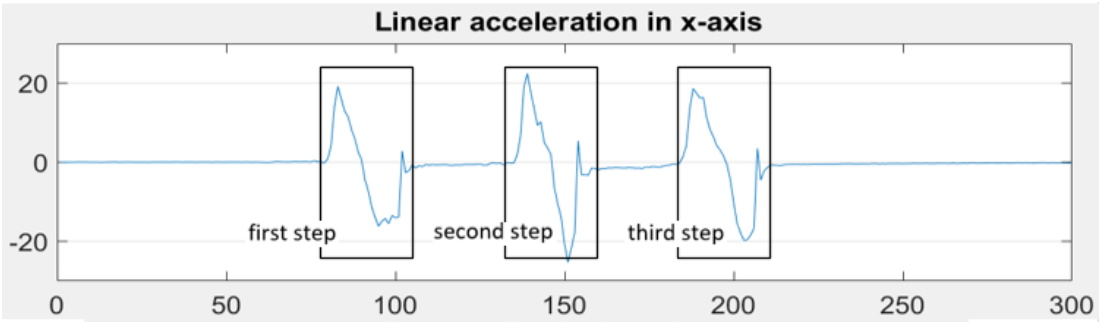



Figure 4.8: Stride determination based on acceleration in x-axis

Moreover, we can also obtain the pattern of gait cycle based on acceleration in x-axis as shown in Figure 4.9.

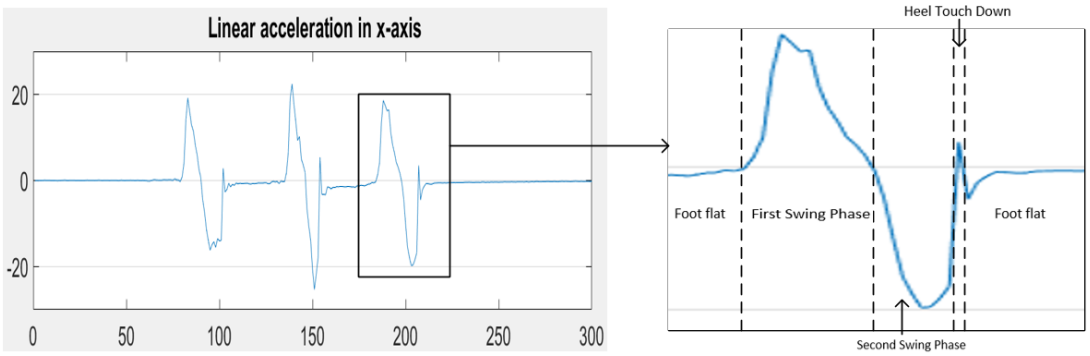



Figure 4.9: Horizontal acceleration (x-axis) signal pattern walking phase

Furthermore, experiments were done to check if Euler angles are related to the stride length although Euler angles are usually used to represent o the orientation of an object. Figure 4.10 shows the changes of Euler angles when walking. It is obvious that the pitch angle (between the heel and the ground) changes regularly. It is also included in the ANN training dataset.

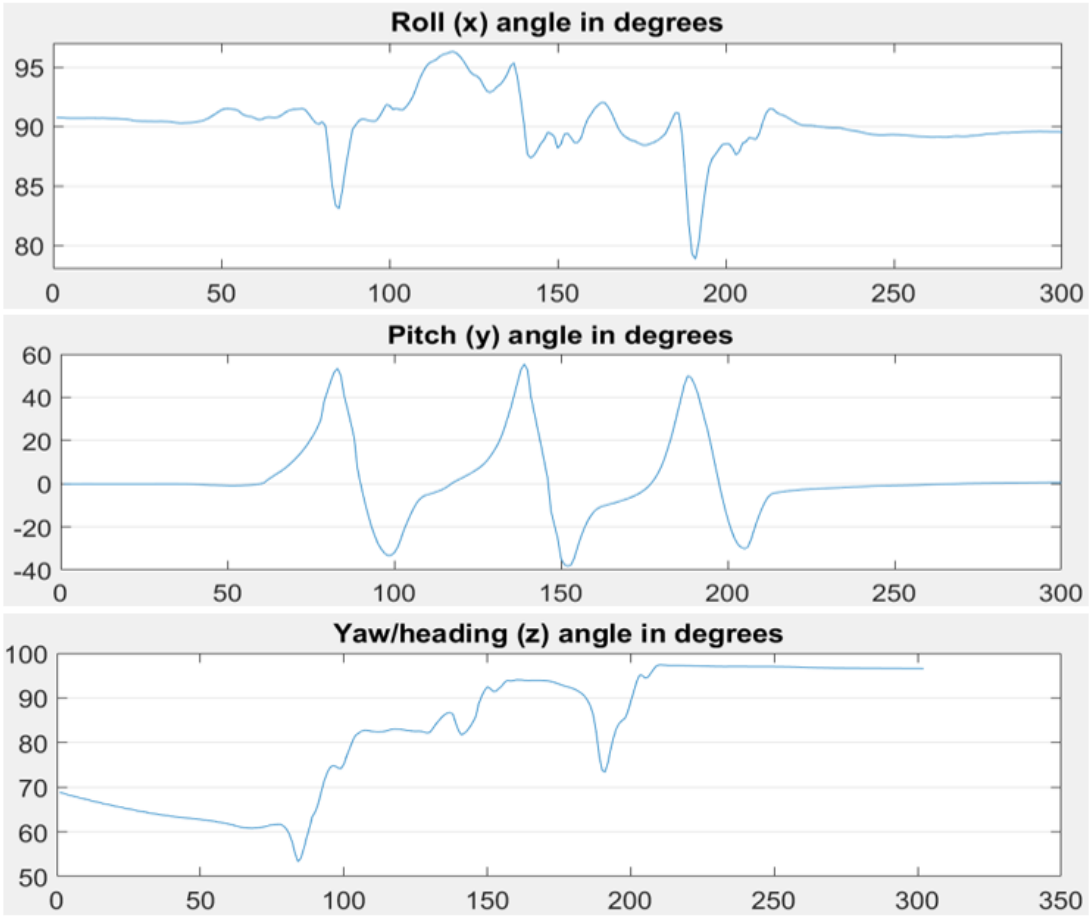



Figure 4.10: Euler angles signals in degrees

In order to remove the noises of the extracted acceleration and Euler angle data, the noise filtering technique is applied. As the movement of the body change irregularly in every step, the acceleration measurements from the sensor contain some unexpected error such as large spikes, They are manually removed before the data processing. Finally, 930 samples were used for ANN training:

- 310 sample for class 1 ($120 \pm 5cm$).
- 310 sample for class 2 ($130 \pm 5cm$).
- 310 sample for class 3 ($140 \pm 5cm$).

## 4.4 ANN fitting training dataset

Similar to data collection for ANN classification training, the sensor is running in 40Hz and the collecting duration is 10 seconds. Therefore, each data file contains approximately 400 data points at an interval of 0.025s. Linear time-stamp, acceleration, Euler angles, quaternion and rotation matrix from IMU were collected in real-time and were written these information into Excel files (as shown in Table 4.4).

Table 4.4: Interpretation of ANN fitting data

| | | | |
|---|---|---|---|
| Column 1 | Acceleration in the sensor x axis | Column 7 | Quaternion e element |
| Column 2 | Acceleration in the sensor y axis | Column 8 | Quaternion x element |
| Column 3 | Acceleration in the sensor z axis | Column 9 | Quaternion y element |
| Column 4 | Roll (x) angle in degree | Column 10 | Quaternion z element |
| Column 5 | Pitch (y) angle in degree | Column 11-20 | Rotation matrix in row-major order |
| Column 6 | Yaw/heading (z) angle in degree | | |

In order to collect the data for ANN fitting training dataset, the sensor was taped on the right foot. Then, we walked in normal gait to leave the footprints of every step by using flour (take five steps at a time), as shown in Figure 4.11. The distance from heel to heel will be measured and it will be used as the target in ANN fitting training dataset. Similar to the training dataset for ANN classification, the acceleration and Euler angles were collected as shown in Figure 4.12 and Figure 4.13.

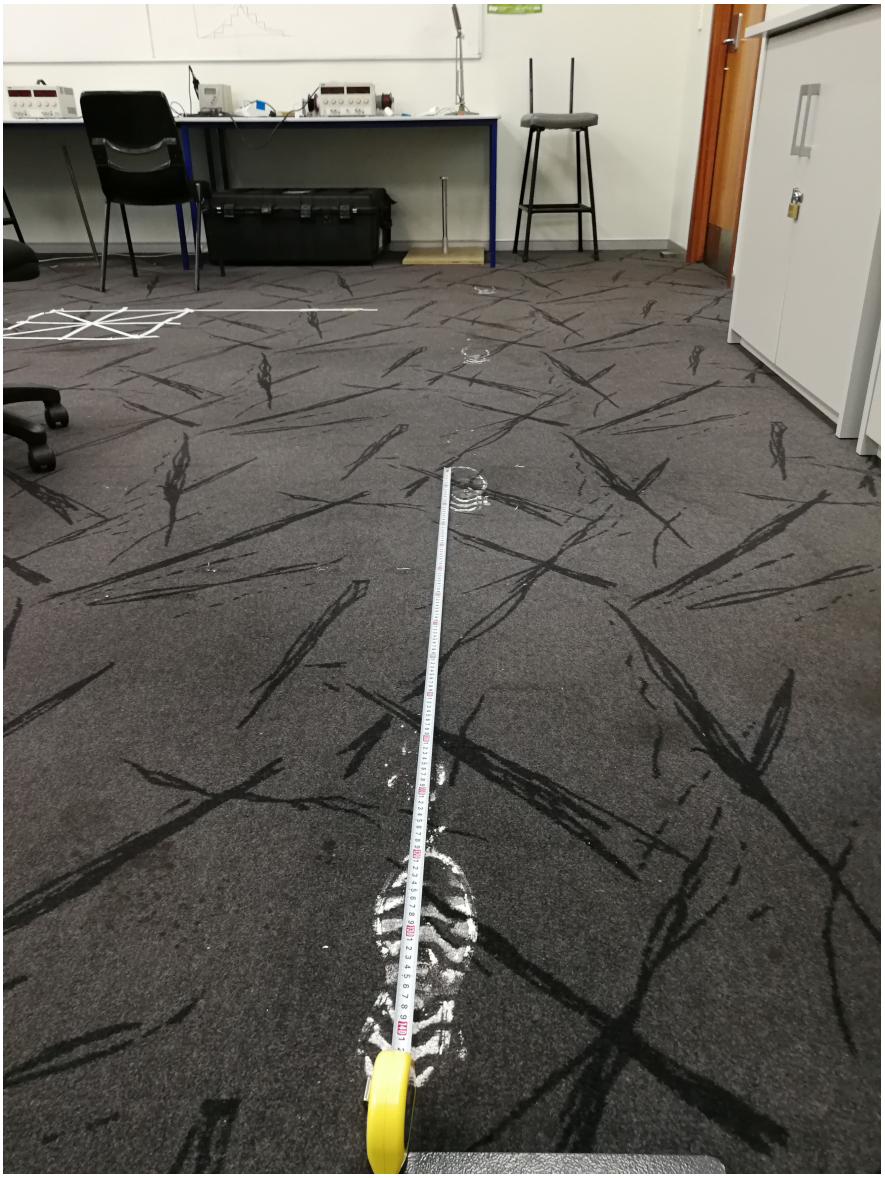


Figure 4.11: Measurement of step length for ANN fitting training data

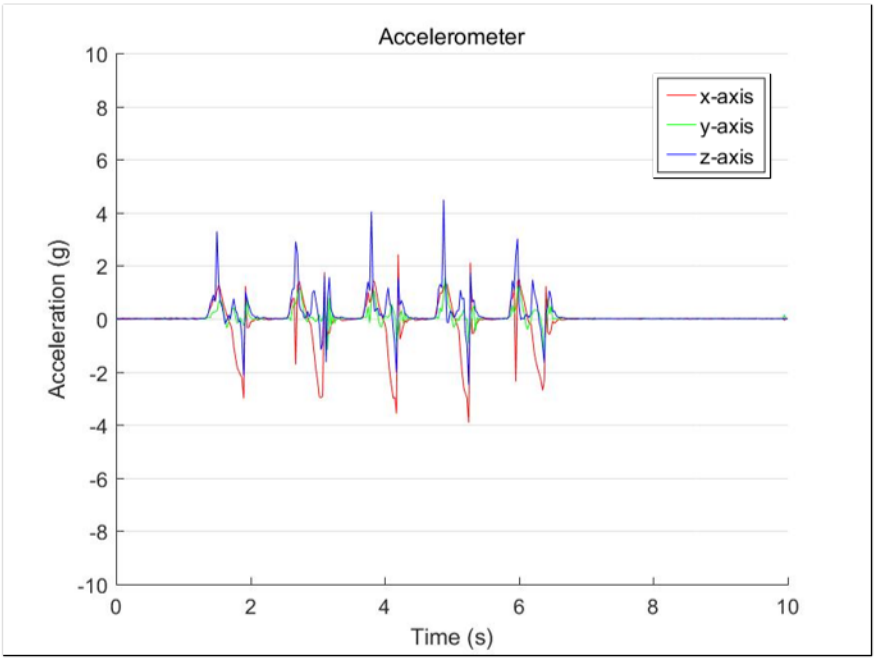



Figure 4.12: Acceleration signal for ANN fitting

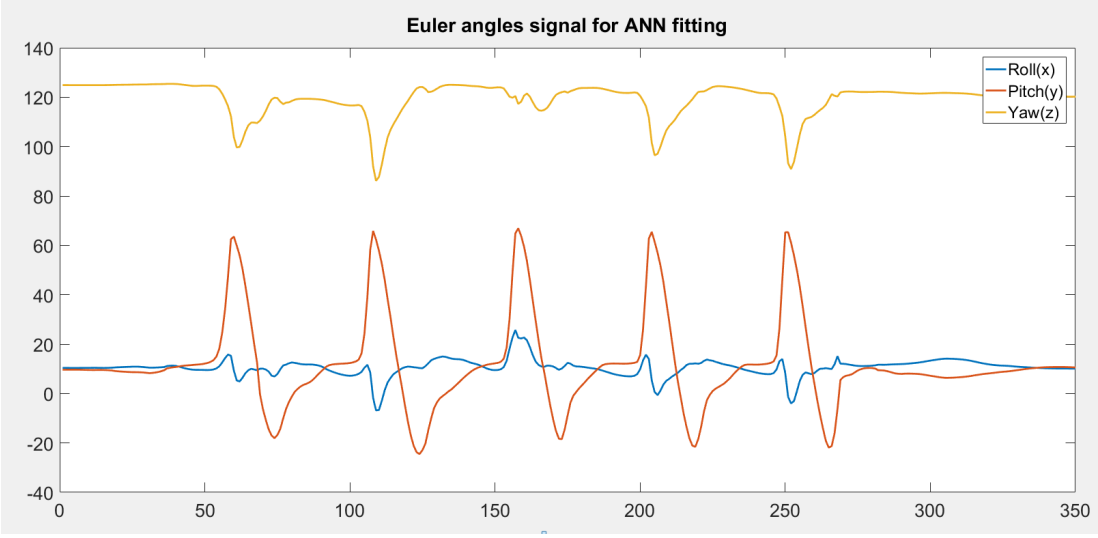



Figure 4.13: Euler angles signal for ANN fitting

Overall 450 input data and corresponding 450 target data were collected for ANN fitting training

## 4.5 Summary

This chapter summarizes the data collection for ANN training through our experiments. All collected data are for the walking in the normal gait. Overall, 400 paired samples in the ANN fitting training dataset and 900 samples (300 samples in each class) in the ANN classification training dataset were collected. Moreover, 50 samples and 30 samples were collected for ANN fitting training dataset and ANN classification dataset respectively as reserve data.

# Chapter 5

# Data processing

## 5.1 Data filtering

As the IMU was affected by the movement of the body, the data may contain several spikes. As these spikes data will affect the ANN training performance, especially the detection of stride via x-axis acceleration data. The spikes needs to be removed.

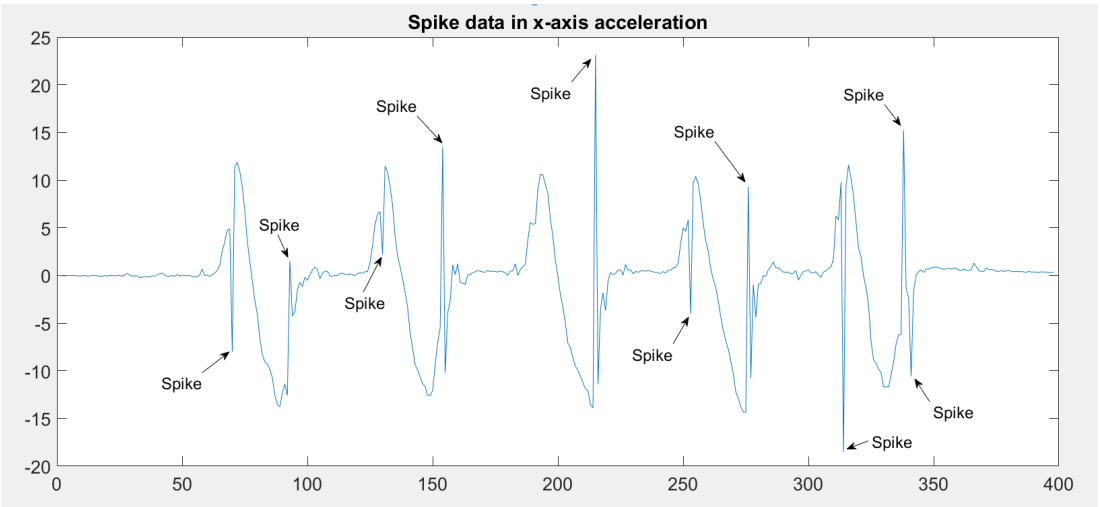
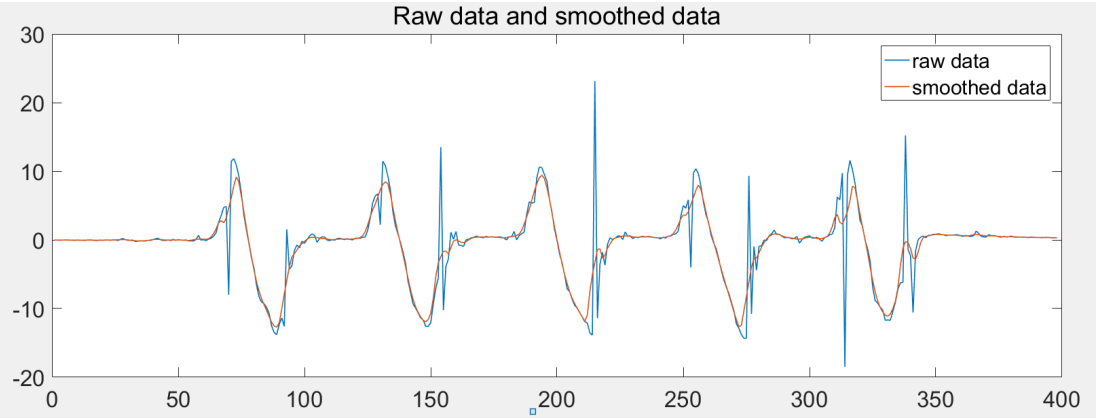



Figure 5.1: Spike data in x-axis acceleration

In our experiment, we use local weight regression method with $SPAN = 5$ and Savitzky-Golay method to smooth the data. The smoothed data processed by these two

filtering methods are shown in Figure 5.2.

[Data filtering using local weight regression method]




[Data filtering using Savitzky-Golay method]

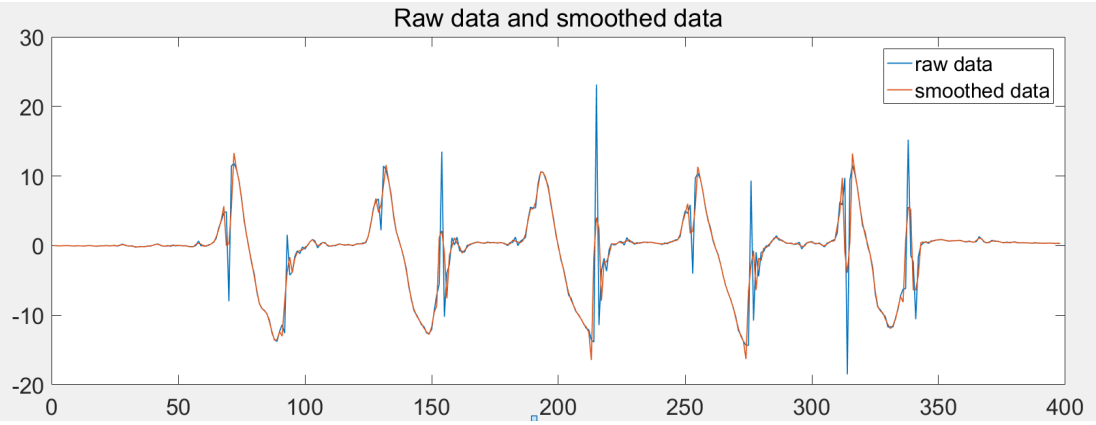



Figure 5.2: Two filtering methods

By comparing these two methods, we found that local weight regression method have better performance because it removed the spike as we expected and also provides smoother data. As for the Savitzky-Golay method, it reduced the spike, but the smoothed data still contains several small spikes as well as signal noise.

## 5.2 Gait detection

The gait detection is an essential part of stride length estimation. In this part, we extract each step from the smoothed data. Then, we record the time-stamp of the start point and the end point of each step. By recording the start and end point time stamp, it can also be used to calculate the frequency of each stride. Because the sensor was attached on the right footwear, it sits still on the ground when the left foot moved, the data will remain stationary during the movement of left foot. The data during this period will be ignored. By dividing the data into strides, it is easier for extracting the features for ANN training dataset.

In our experiment, we detect the gait via x-axis linear acceleration using zero crossing algorithm. Moreover, we add several determine conditions to eliminate the influence of the sensor noise. The detailed process is described in Figure 5.3:

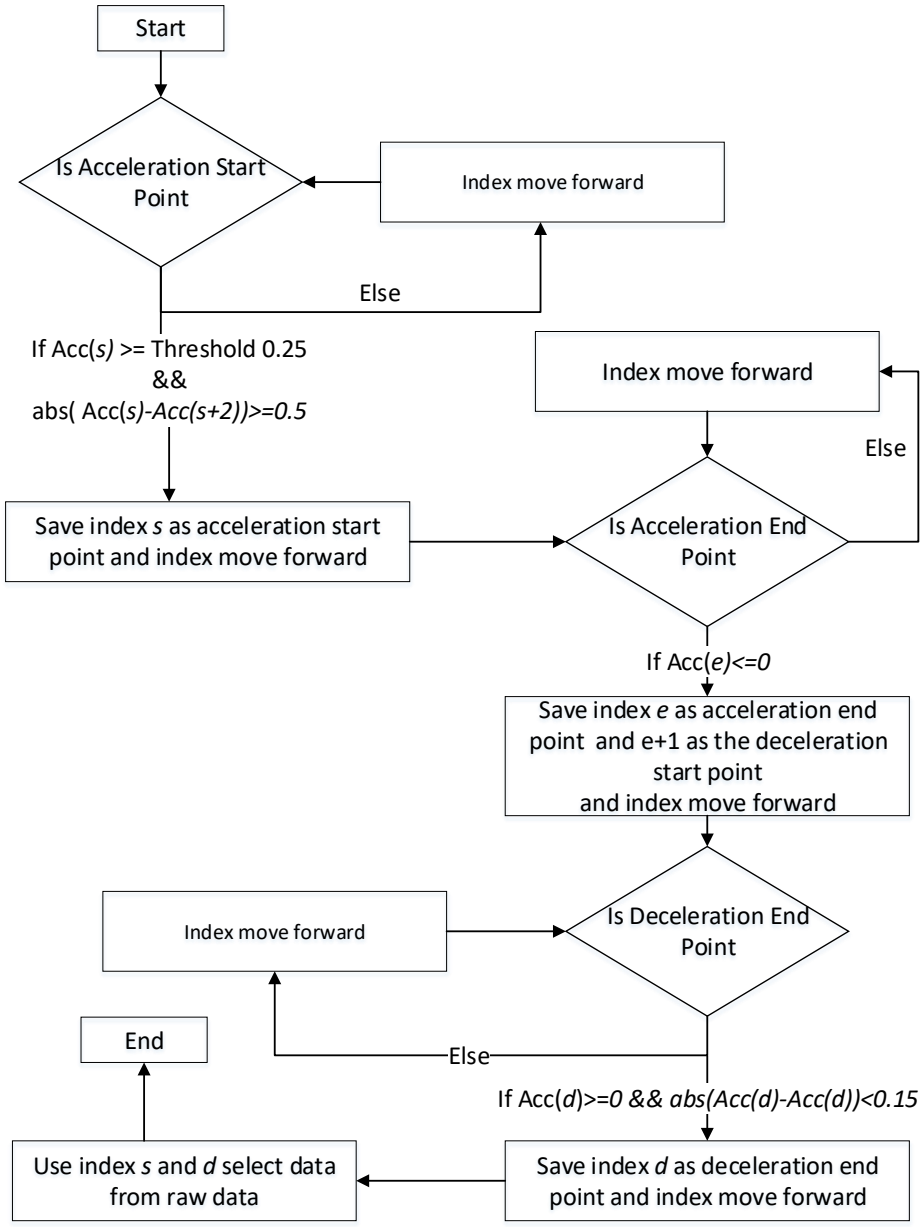



Figure 5.3: Step detection process

When the sensor is stationary, the signal output from the IMU usually has a small offset and the average of the offset is about 0.25. Therefore we set up a threshold to remove the offset. The expression $abs(Acc(s) - Acc(s+2)) >= 0.5$ is to determine that the acceleration is actually rising. For example, the result of one gait detection shown as below.

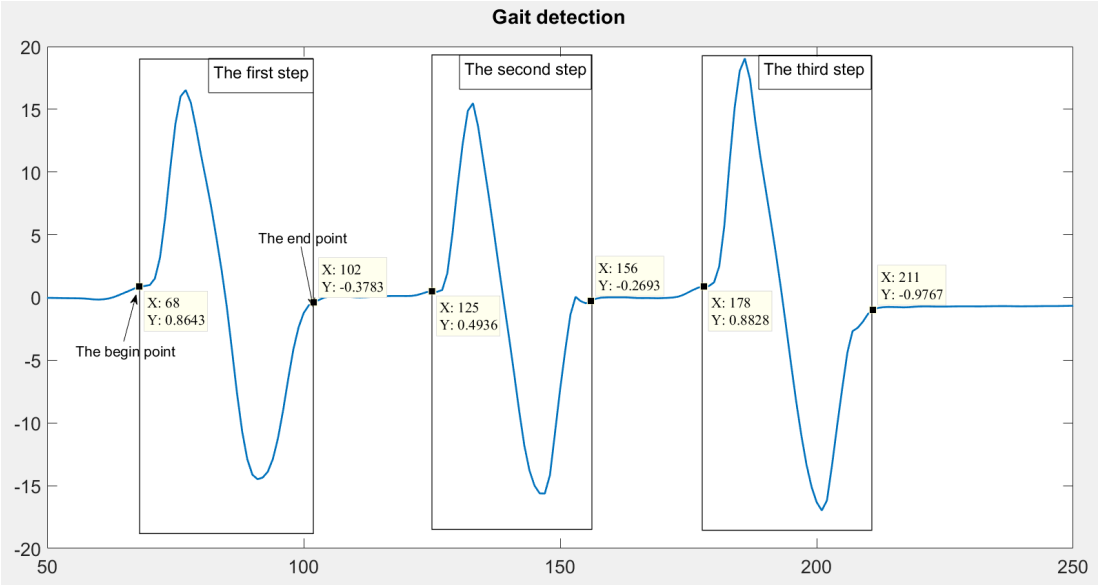



Figure 5.4: Gait detection

## 5.3 Feature selection

First of all, we need to select features for ANN training. We use similar methods to extract the features for ANN classification and fitting data. The acceleration is considered to be an important feature for the stride length estimation. Therefore, we selected the maximum and minimum of the acceleration data in x-axis as the features. Also, we calculated the time period of the acceleration is positive and the time period of when acceleration is negative as the features as shown in Figure 5.5.

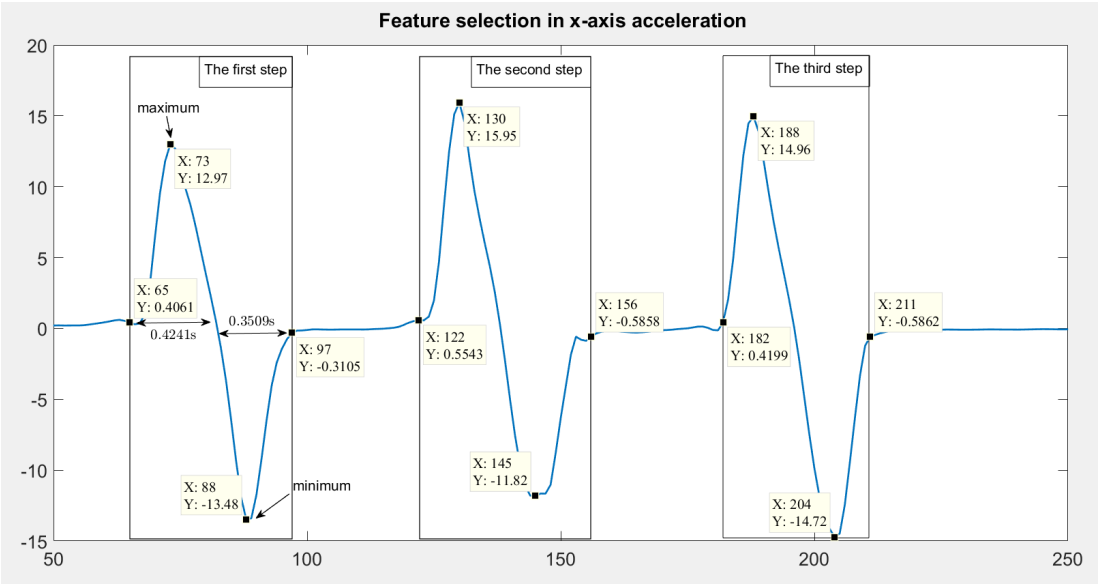



Figure 5.5: Feature selection in x-axis acceleration

In our experiments, we combined the acceleration data and Euler angles data to estimate the stride length. We believe that the maximum and minimum of the pitch angle is also the vital features to classify the three classes of stride length. We use same time-stamp of maximum and minimum acceleration to locate the maximum and minimum pitch angle of each step (Figure 5.6).

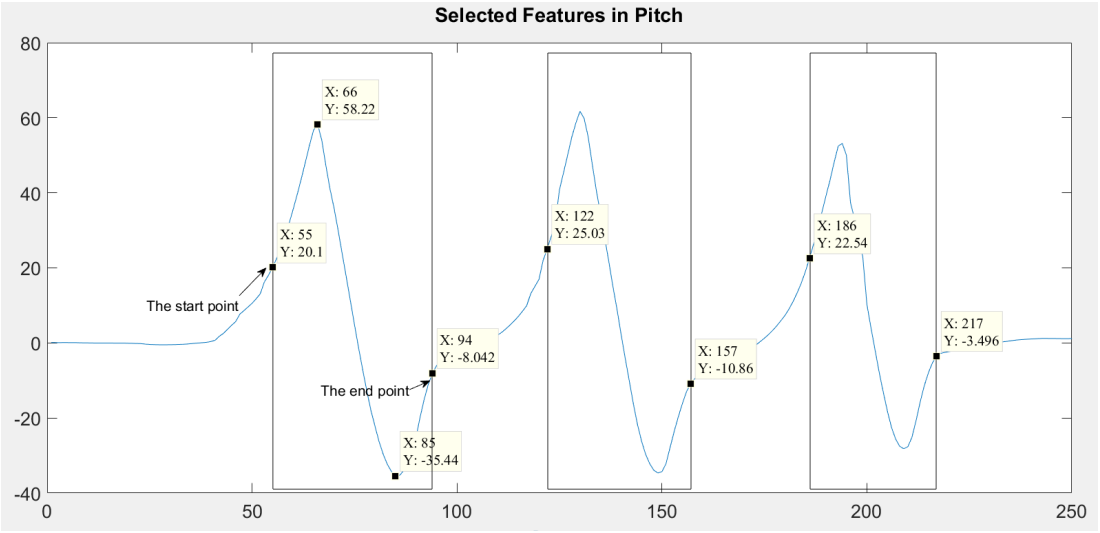



Figure 5.6: Selected features in Pitch angle

Interpretation of the input data for ANN fitting and classification shows in Table 5.2 and Table 5.1 respectively.

Table 5.1: Interpretation of the input data for ANN fitting

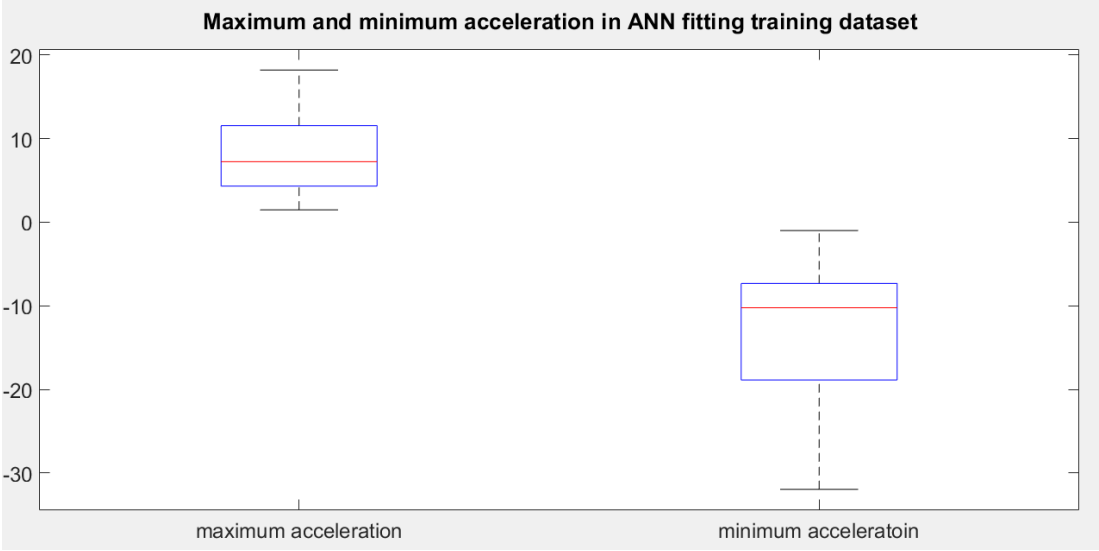
| Column 1 | Column 2 |
|---|---|
| Acceleration peak | Deceleration peak |
| Column 3 | Column 4 |
| Acceleration time | Decceleration time |

Table 5.2: Interpretation of the input data for ANN classification

| Column 1 | Column 2 |
|---|---|
| Acceleration peak | Deceleration peak |
| Column 3 | Column 4 |
| Acceleration time | Deceleration time |
| Column 5 | Column 6 |
| The maximum of the pitch angle | The minimum of the pitch angle |

In terms of ANN fitting dataset, as shown in Figure 5.7 , the box plot displays 4 features we used for ANN fitting training dataset. According to Figure 5.7, there are 23 outliers in the acceleration time data and 8 outliers in the deceleration time data. We removed total 31 samples from the ANN fitting training dataset.

[Boxplot of maximum and minimum of the acceleration in ANN fitting training dataset]




[Boxplot of acceleration time and deceleration time in ANN fitting training dataset]

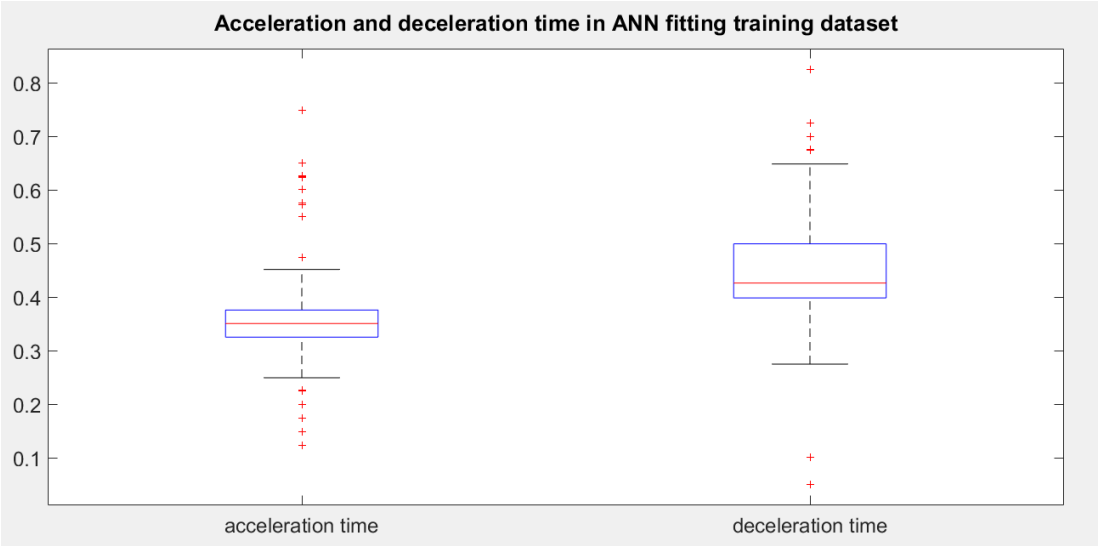
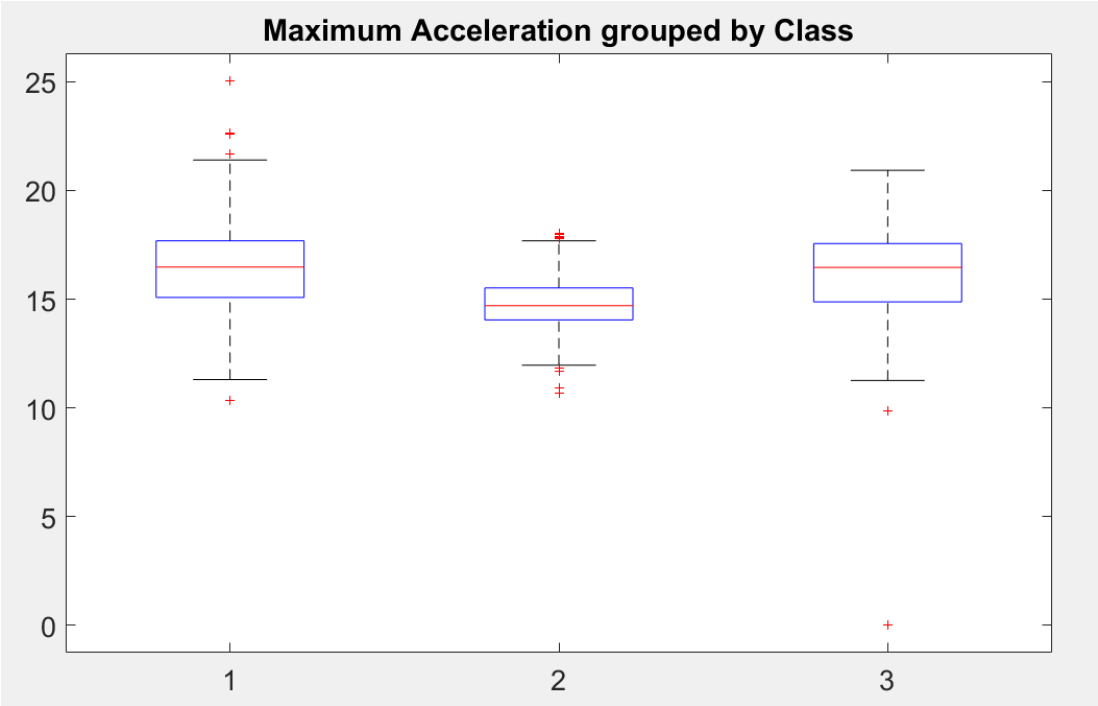



Figure 5.7: Boxplot of four features in ANN fitting training dataset

In terms of ANN classification dataset, we can display the maximum and minimum acceleration of each step in box plot to exam the distribution and difference between these three classes in ANN classification training dataset(Figure 5.8). Each box visually represents the maximum and minimum acceleration data of each sample from three

classes.

[Boxplot of Maximum acceleration grouped by Class]




[Boxplot of Minimum acceleration grouped by Class]

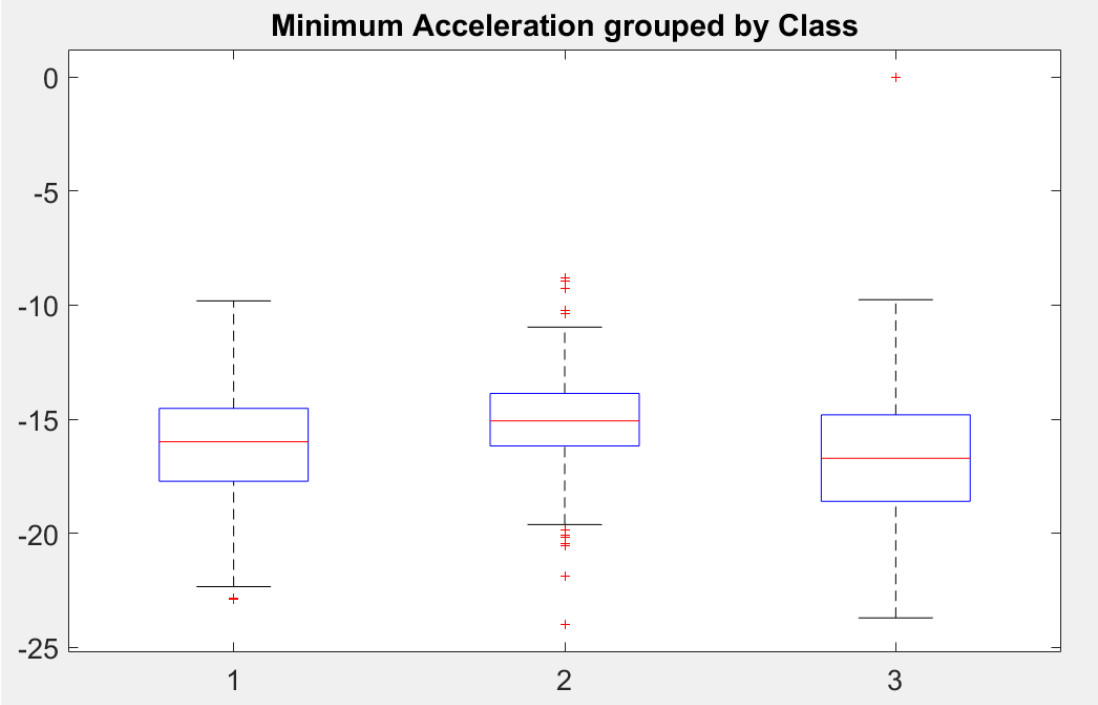



Figure 5.8: Boxplot of acceleration grouped by Class

For example, as the box plot of class 1 shown in the first figure in Figure 5.8, the median value was 16.4768 while the maximum and the minimum value were 25.058$m/s1$ and 10.3452$m/s$ respectively. There were also five outliers in Class one, we will remove the outliers from the ANN training data. We extract the information from these two figure into Table 5.3 and Table 5.4.

Table 5.3: The Overview of Maximum Acceleration grouped by Class

| | Class 1 | Class 2 | Class 3 |
|---|---|---|---|
| Median | 16.4768 | 14.701 | 16.4585 |
| Maximum | 25.0581 | 18.0318 | 20.9259 |
| Minimum | 10.3452 | 10.7017 | 0 |
| 25th Percentile | 15.0804 | 14.0463 | 14.8775 |
| 75th Percentile | 17.6831 | 15.5191 | 17.5619 |
| Num Points | 300 | 300 | 300 |
| Num Finite Outliers | 5 | 10 | 4 |

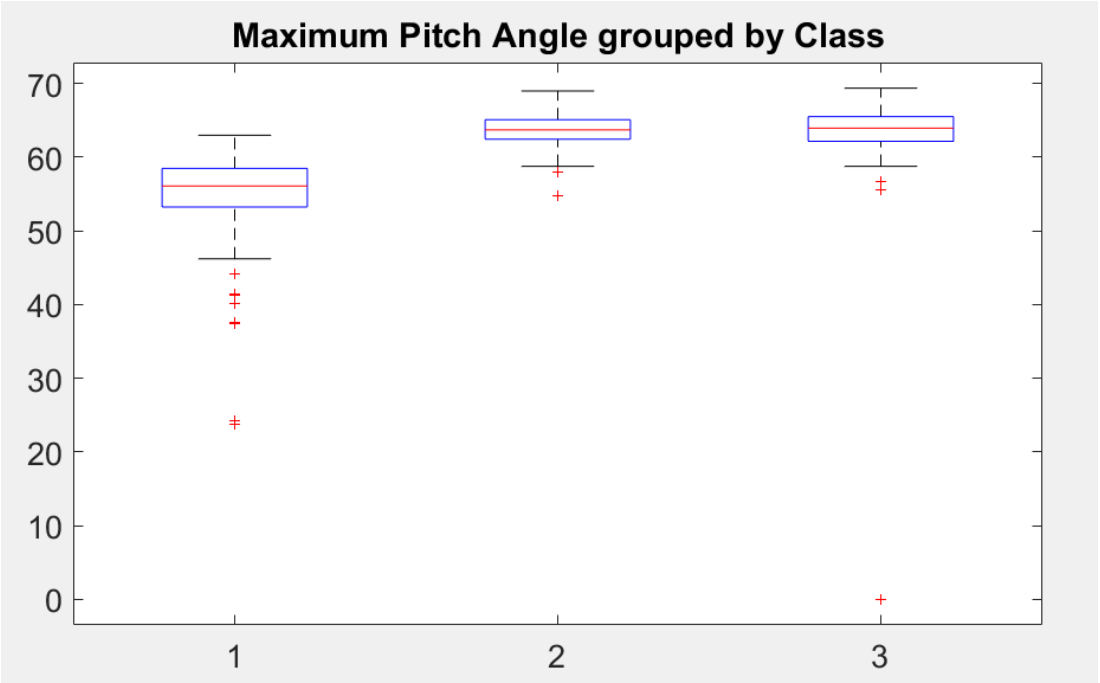
Table 5.4: The Overview of Minimum Acceleration grouped by Class

| | Class 1 | Class 2 | Class 3 |
|---|---|---|---|
| Median | -15.9864 | -15.0672 | -16.7091 |
| Maximum | -9.8143 | -8.8112 | 0 |
| Minimum | -22.8677 | -24.0086 | -23.6998 |
| 25th Percentile | -17.7159 | -16.1705 | -18.5952 |
| 75th Percentile | -14.5243 | -13.867 | -14.8049 |
| Num Points | 300 | 300 | 300 |
| Num Finite Outliers | 2 | 12 | 3 |

Another important feature that we used for the ANN classification is the pitch angle. As shown in Figure 5.9. Then, we plot the maximum and minimum of the pitch angle grouped by class in box plot to observe the distribution of each class.

For example, as the box plot of class 1 shown the first figure in Figure 5.9, the median value of Pitch angle was 56.0728 while the maximum and the minimum value were 62.9733 degree and 23.7458 degrees respectively. We extract the information from these two figure into Table 5.5 and Table 5.6.

[Boxplot of Maximum Pitch Angle grouped by Class]




[Boxplot of Minimum Pitch Angle grouped by Class]

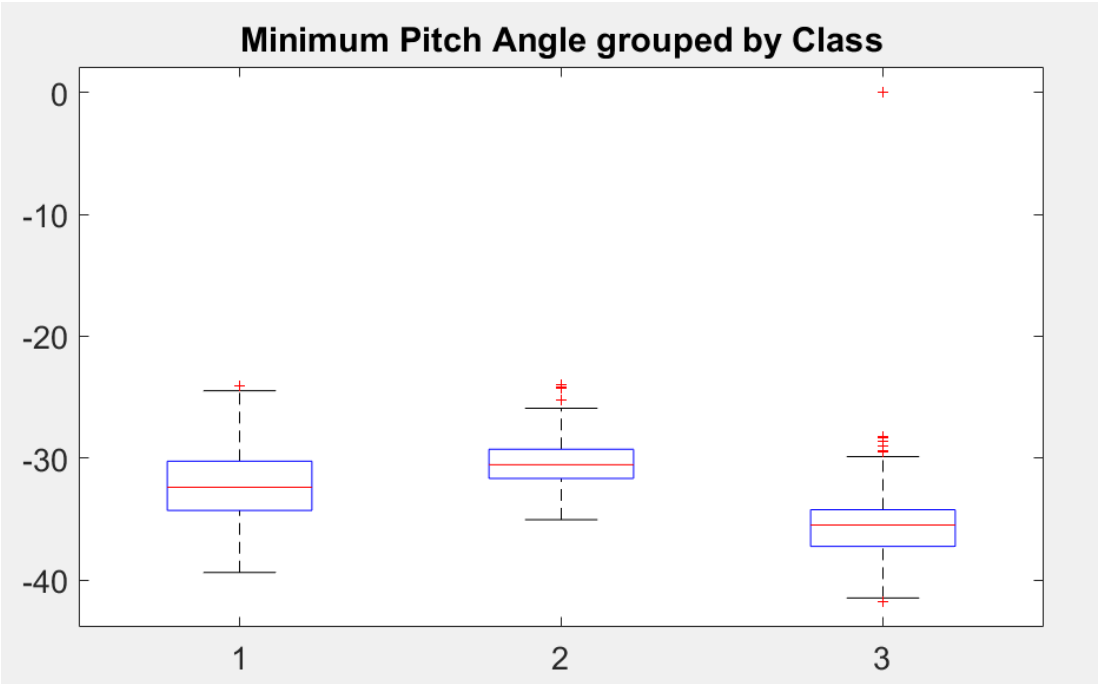



Figure 5.9: Boxplot of Pitch Angle grouped by Class

The information of these "box" can be sorted into Table 5.5 and Table 5.6.

Table 5.5: The Overview of Maximum Pitch Angle grouped by Class

| | Class 1 | Class 2 | Class 3 |
|---|---|---|---|
| Median | 56.0728 | 63.6922 | 63.9389 |
| Maximum | 62.9733 | 68.9665 | 69.3559 |
| Minimum | 23.7458 | 54.825 | 0 |
| 25th Percentile | 53.2278 | 62.4171 | 62.1331 |
| 75th Percentile | 58.4779 | 65.0689 | 65.4865 |
| Num Points | 300 | 300 | 300 |
| Num Finite Outliers | 9 | 2 | 5 |

Table 5.6: The Overview of Minimum Pitch Angle grouped by Class

| | Class 1 | Class 2 | Class 3 |
|---|---|---|---|
| Median | -32.3809 | -30.5344 | -35.4774 |
| Maximum | -24.0773 | -24.0099 | 0 |
| Minimum | -39.3794 | -35.0516 | -41.7581 |
| 25th Percentile | -34.2832 | -31.6586 | -37.2281 |
| 75th Percentile | -30.2522 | -29.2664 | -34.2277 |
| Num Points | 300 | 300 | 300 |
| Num Finite Outliers | 1 | 4 | 10 |

Finally, after removed the outliers, a dataset of 400 samples with four features was created for ANN fitting and a dataset of 900 samples with 6 features (300 samples for each classes) was created for ANN classification.

## 5.4 Summary

In this section, we used the local weight regression method and Savitzky-Golay method to filter the data collection at first. Then the filtered data was used to extract each gait by using the zero crossing method. Moreover, we selected four features for ANN fitting training, namely, acceleration peak, deceleration peak, acceleration time and deceleration time; and six features for ANN classification training, namely, acceleration peak, deceleration peak, acceleration time, deceleration time, the maximum of the pitch angle and the minimum of the pitch angle. After data processing, the training data set for ANN fitting consists of 400 samples with 4 features and the training dataset for ANN classification consists of 900 samples with 6 features.

# Chapter 6

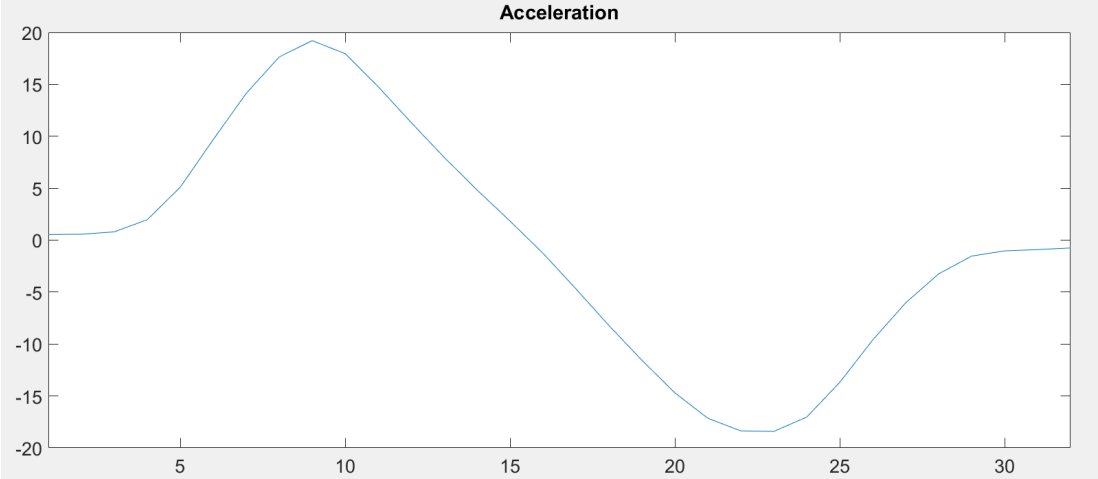
# Results and Discussions

## 6.1 Step length estimation using double integration of acceleration

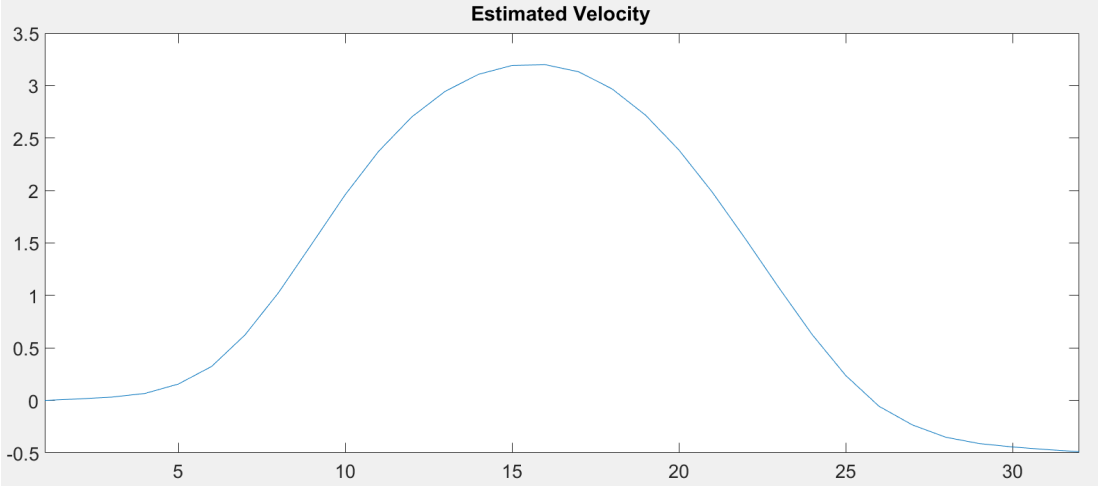
One of the traditional methods of the stride length estimation is using double integration. In double integration method, the velocity can be obtained by integration of the acceleration data, and the displacement of the position can be obtained by integration of the velocity. In this part, we estimate the distance within the gait cycle by using the trapezoidal rule and Simpson's rule to integrate the acceleration data.

We use the acceleration data in Class 2 ($130 \pm 5cm$) of the dataset for ANN classification to testing the performance of the double integration method. Firstly, we applied the trapezoidal rule to the filtered and smoothed acceleration data (the dataset collected for ANN classification training ). As shown in Figure 6.1, we tested samples from each class and found that the performance is not satisfactory with the error is more than 20%. When using the trapezoidal rule, when the pedestrian stands still, the obtained velocity is not $0$ and the position is still drafting.

[IMU Recorded Acceleration $(meter/second^2)$]




[Estimated Velocity using trapezoidal rule $(meter/second)$]




[Estimated Position using trapezoidal rule $(meter)$]

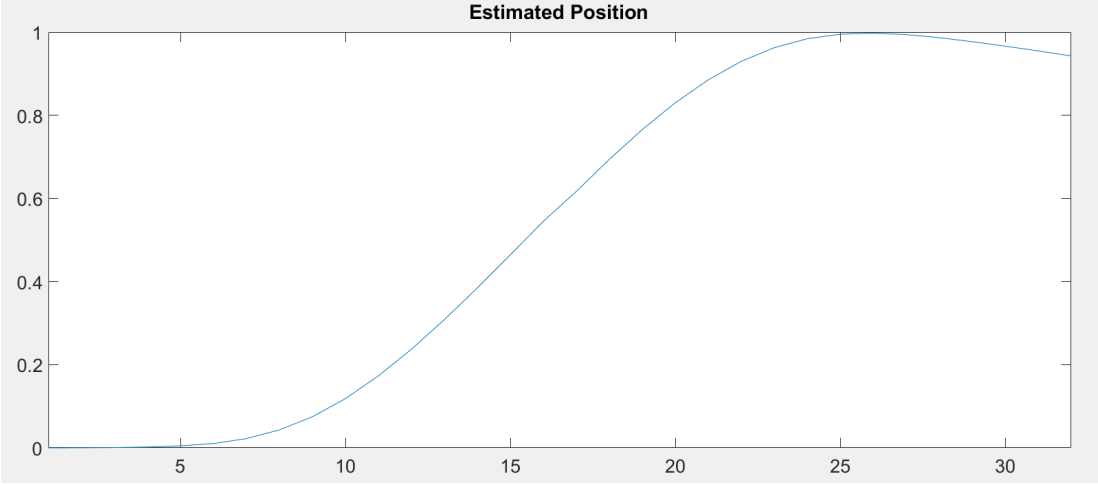
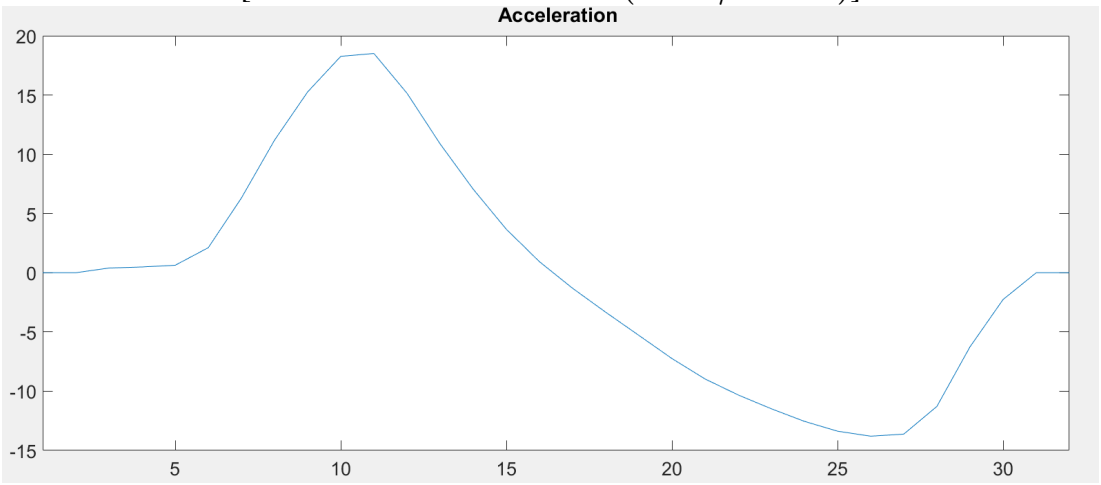


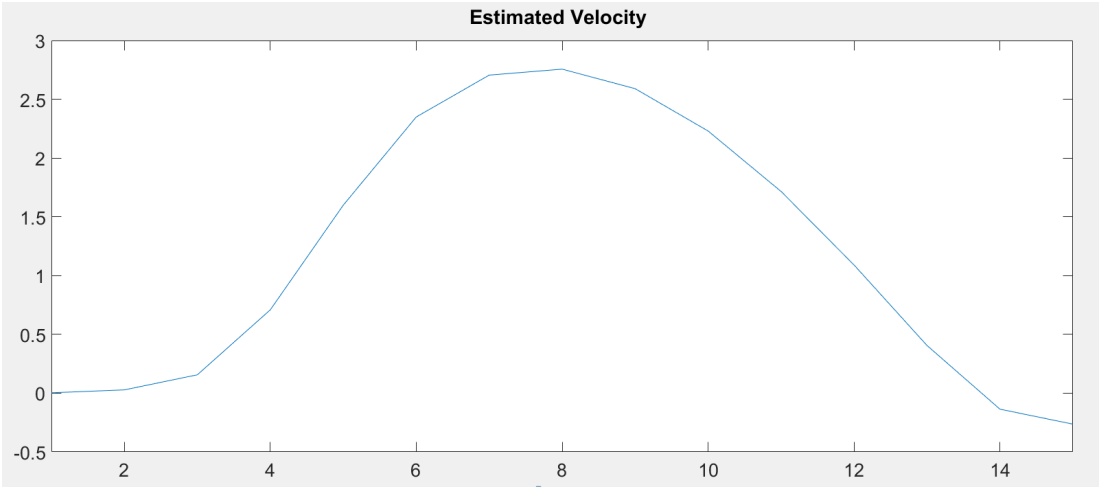

Figure 6.1: Double integration using trapezoidal rule

Better results are got when the Simpson's Rule is applied (e,g. estimated velocity is close to $0$ when the pedestrian does not walk and the estimated position does not drift) However, the accuracy of measurement is still unsatisfactory. As shown in Figure 6.2, the error is still significant

[IMU Recorded Acceleration $(meter/second^2)$]




[[Estimated Velocity using Simpson's Rule $(meter/second)$]




[Estimated Position using Simpson's Rule $(meter)$]

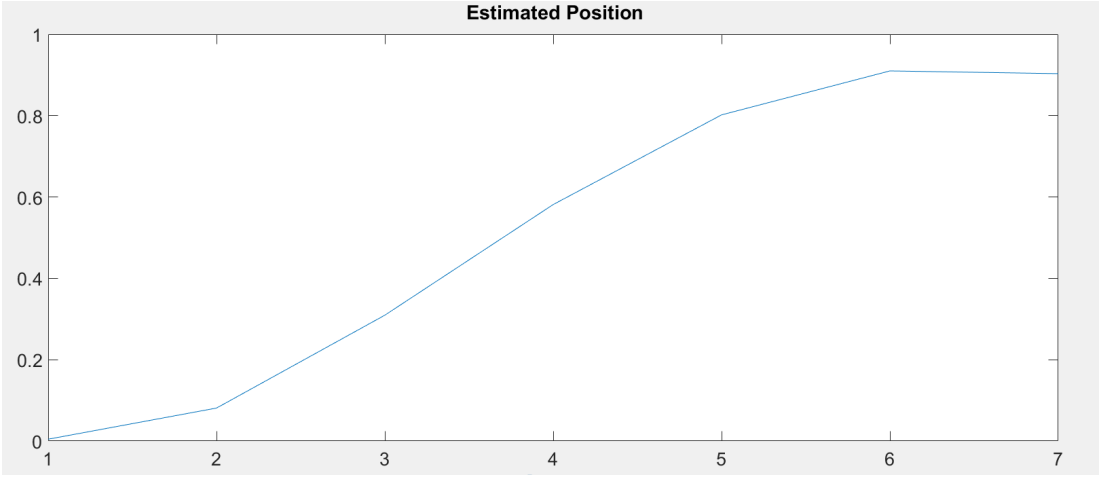



Figure 6.2: Double integration using Simpson's Rule

Due to the nature of the human gait, the feet will continuously rotate in Y-axis, which will cause significant noise in the acceleration recording. As shown in Figure 6.3, the target in class 2 is $130 \pm 5cm$. However,the mean value of estimated stride length by Simpson's Rule is about 100cm and vary in a range of 30cm.

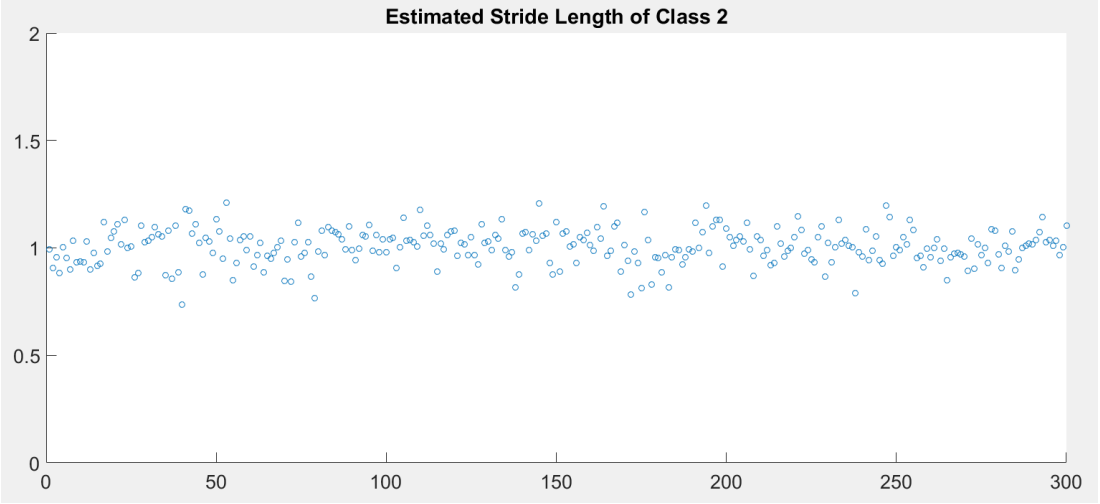



Figure 6.3: Estimated stride length by Using Simpson's Rule

## 6.2 Step length estimation using ANN fitting function

We adopted the ANN fitting function for step length estimation. Three different training algorithms in the MATLAB neural network toolbox, namely Bayesian regularization, Levenberg-Marquardt and Scaled Conjugate Gradient training algorithm are applied (Demuth, Beale & Hagan, 2008). We use 70% of the training data for training, 15% for validation and 15% for testing, as shown in Figure 6.4. The testing and validation dataset has same features as the training dataset, but are not contained in the training dataset. The training performance will be measured by mean squared error (MSE).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{x}_i - x_i)^2 \tag{6.1}$$

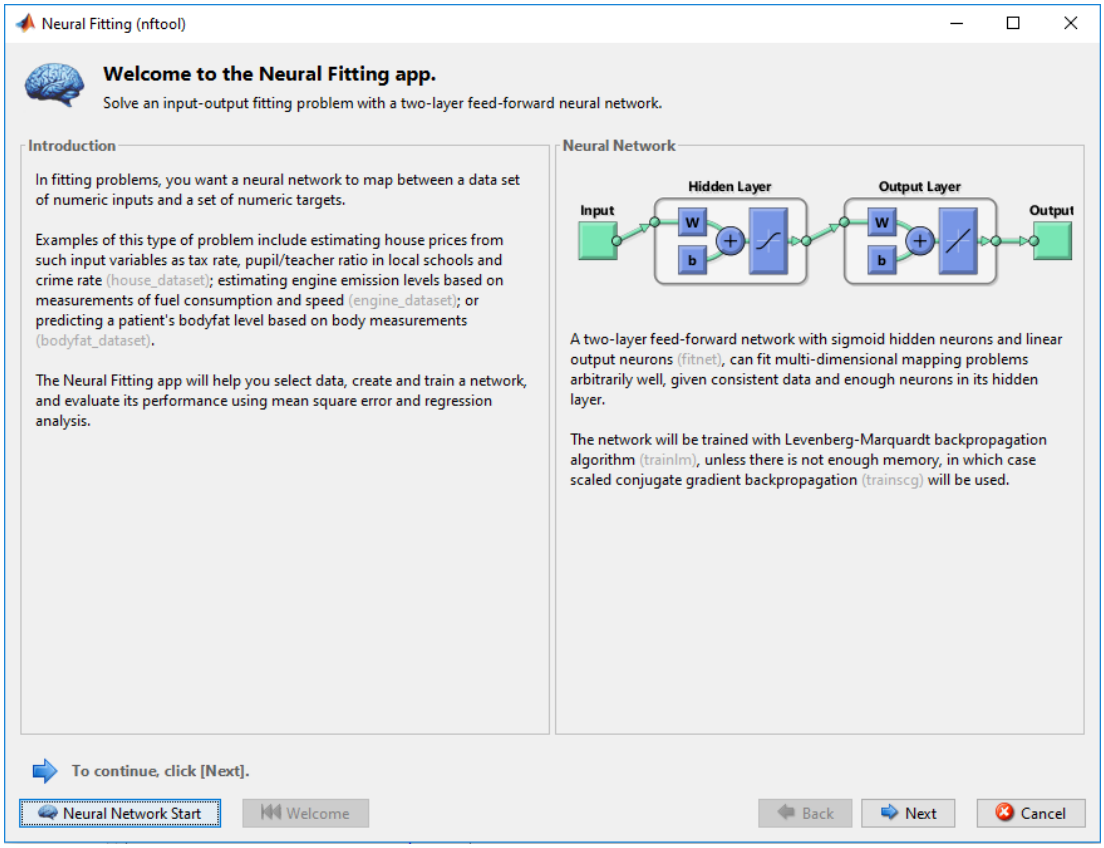



Figure 6.4: ANN fitting tool box

The input dataset for the ANN training is as shown in Figure 6.5. Column One and Column Two are the maximum and minimum values of the acceleration data of each step; the last two columns are the acceleration period and deceleration period respectively. The input dataset is a 400*4 matrix, representing 400 samples with four features.

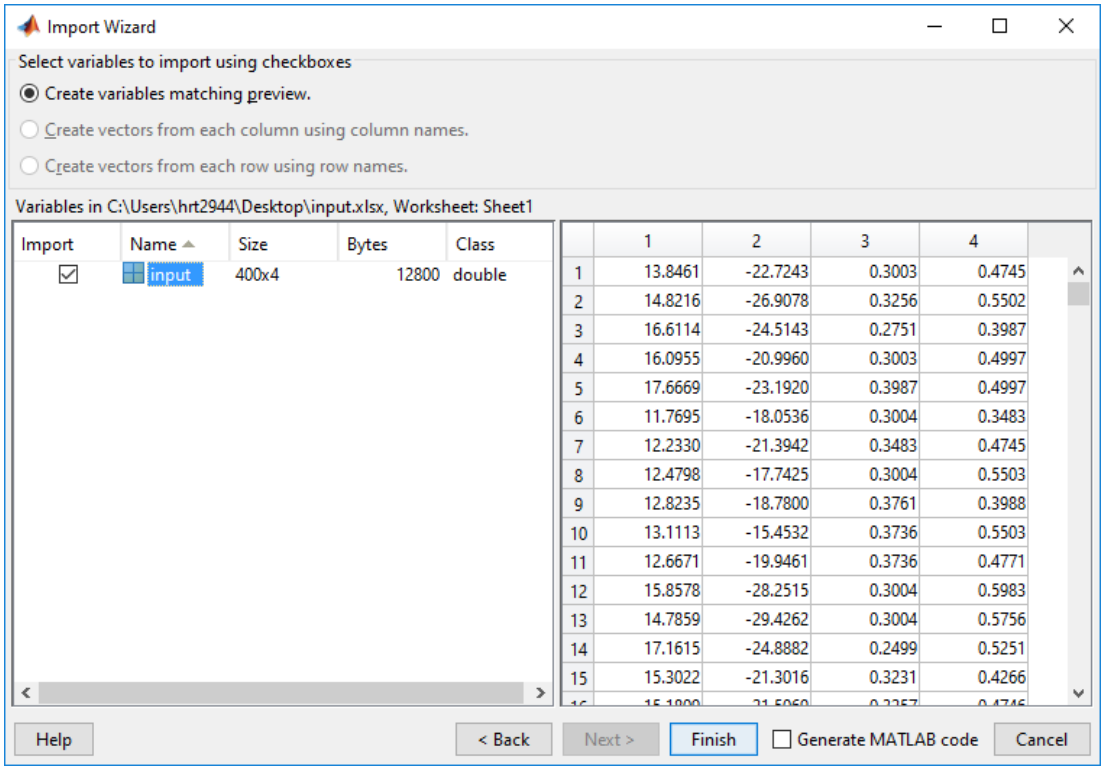



Figure 6.5: Input dataset for ANN fitting

### 6.2.1 Parameter Optimization for ANN Fitting

#### Levenberg-Marquardt backpropagation

An example of the error between the target data set and output data set (ANN fitting result) using 10 neurons are shown in Figure 6.6.

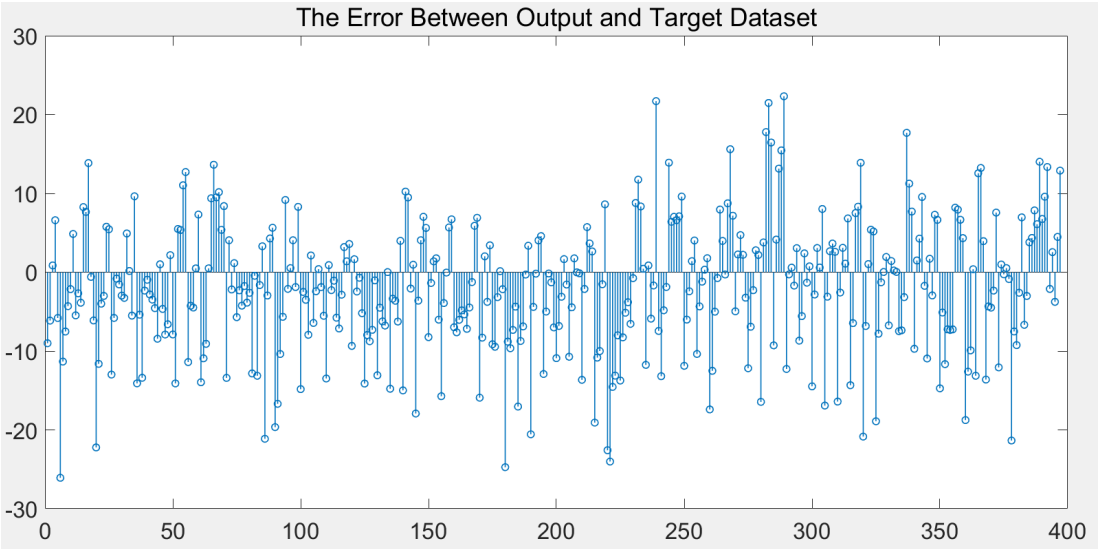



Figure 6.6: The error between target and output dataset (unit: cm)

Firstly, we use the Levenberg-Marquardt training algorithm for ANN fitting. Table 6.1 shows the average results in ANN fitting.

Table 6.1: Levenberg-Marquardt (LM) with Different Neuron Number

| Neuron Number | MSE (Training data) | MSE (Testing data) | Variance |
|---|---|---|---|
| 1 | 75.8945 | 76.9167 | 76.3616 |
| 5 | 48.1701 | 68.2068 | 67.2133 |
| 10 | 54.3578 | 61.0136 | 56.2246 |
| 30 | 42.6967 | 50.4028 | 53.3877 |
| 50 | 39.6075 | 66.0675 | 47.2345 |
| 100 | 33.3322 | 70.0772 | 49.0071 |
| 150 | 22.6448 | 79.2244 | 49.9281 |

As shown in Table 6.1, with the LM training algorithm, the MSE of the result of the test with the training data is decreasing when the neuron number increases. However,

the MSE of the result of fitting testing data, which not included in the training dataset shown a different trend. When the neuron is increased from 1 to 30, the MSE of the testing data decreases; then it increases when the neuron number is increased from 30 to 150, which indicated the over-fitting.

### Scaled Conjugate Gradient backpropagation

Secondly, the performance of ANN estimation using Scaled Conjugate Gradient algorithm is shown in Table 6.2.

Table 6.2: Scaled Conjugate Gradient (SCG) with Different Neuron Number

| Neuron Number | MSE (Training data) | MSE (Testing data) | Variance |
|---|---|---|---|
| 1 | 74.7883 | 92.224 | 78.5312 |
| 5 | 71.6541 | 79.8318 | 75.789 |
| 10 | 64.0383 | 88.3818 | 66.7949 |
| 30 | 59.1804 | 76.4689 | 62.6881 |
| 50 | 53.9711 | 47.2422 | 61.1503 |
| 100 | 44.6101 | 74.7196 | 55.7348 |
| 150 | 45.968 | 89.2161 | 58.9499 |

As shown in Table 6.2, similar to LM training algorithm, after the neuron number is increased to 50, the MSE of the result when using the testing data declined, which indicated that SCG training algorithm is over-fitting. The result shows 50 is the suitable neuron number for our neuron network with SCG training algorithm.

**Bayesian Regularization**

The performance of ANN estimation using Bayesian Regularization training algorithm is shown in Table 6.3.

Table 6.3: Bayesian Regularization (BR) with Different Neuron Number

| Neuron Number | MSE (Training data) | MSE (Testing data) | Variance |
|---|---|---|---|
| 1 | 78.0621 | 77.279 | 78.1303 |
| 5 | 66.8386 | 63.7348 | 66.5378 |
| 10 | 51.4304 | 58.3422 | 51.8495 |
| 30 | 48.7449 | 56.8234 | 50.0574 |
| 50 | 45.9828 | 60.3989 | 48.286 |
| 100 | 45.7741 | 60.3171 | 48.0901 |
| 150 | 45.696 | 86.071 | 51.9237 |

When using the Bayesian Regularization as the training algorithm in ANN, the MSE of the result of fitting the training data and testing data were decreased with the neuron number increase from 1 to 10; then they slightly changed when neuron number is continuously increased. Unlike SCG and LM training algorithms, the MSE of the result of the testing data did not effected by the increment of the neuron number, when over-fitting.

### 6.2.2 The Result of ANN fitting

According to previous experiments, we found the optimal parameters for ANN fitting process is using the Levenberg-Marquardt training algorithm with 30 neurons. Therefore, we use 10 step as one trail, then we used a trained ANN (Levenberg-Marquardt

with 30 hidden neurons) to estimate the step lengths of 50 steps in 5 trails, the result shown in Table 6.4.

Table 6.4: ANN fitting result

| Trail | Target (m) | ANN output (m) | Error (m) | Error (%) |
|---|---|---|---|---|
| 1 | 12.1 | 11.7131 | 0.3869 | 3.1975 |
| 2 | 12.22 | 11.8709 | 0.349 | 2.8560 |
| 3 | 12.58 | 12.4427 | 0.1373 | 1.0914 |
| 4 | 12.58 | 12.2156 | 0.3644 | 2.8967 |
| 5 | 12.44 | 12.1086 | 0.3314 | 2.6640 |
| Overall | 61.92 | 60.3509 | 1.569 | 2.5339 |

As the Table 6.4 shows, the target distance is 61.92 meters and the ANN estimated distance is 60.35 meters, the error between the target and ANN output is less than 2 meters. In particular, the error rate is only 2.53%.

## 6.3 Step length estimation using ANN classification function

In the ANN classification function, we use three different training algorithms in the MATLAB neural network toolbox, namely Scaled Conjugate Gradient, Gradient descent with momentum and adaptive learning rate and Levenberg-Marquardt training algorithm (Demuth et al., 2008). We used the MATLAB MSE function (plotperform) to determine the performance of the ANN. Similar to the ANN fitting function, we use 70% of the training data for training, 15% for validation and 15% for testing, as shown in Figure 6.7.

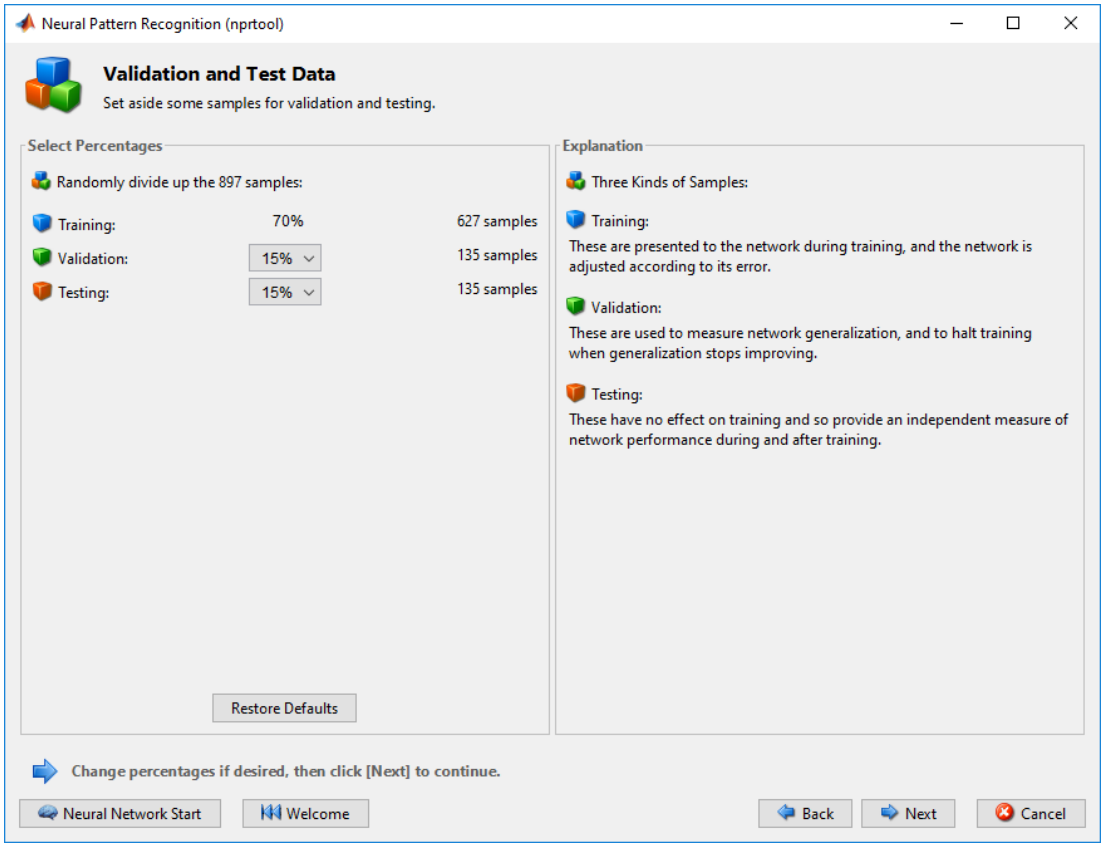



Figure 6.7: ANN classification toolbox

The training stops when any of these conditions occur:

- The epochs reached to the maximum number (1000).
- The time is exceeded the maximum time to train in seconds (3600 seconds).
- The performance gradient falls below the minimum performance gradient (gradient = 0).
- The performance reached the goal (Error = 0).
- The validation performance has increased more than maximum validation failures times since the last time it decreased (maximum validation failures = 6).

The input data set for ANN training is shown in Figure 6.8. The interpretation of each column is shown in Table 5.2. The input dataset is a $900 * 6$ matrix, that represents 900 samples with 6 features.

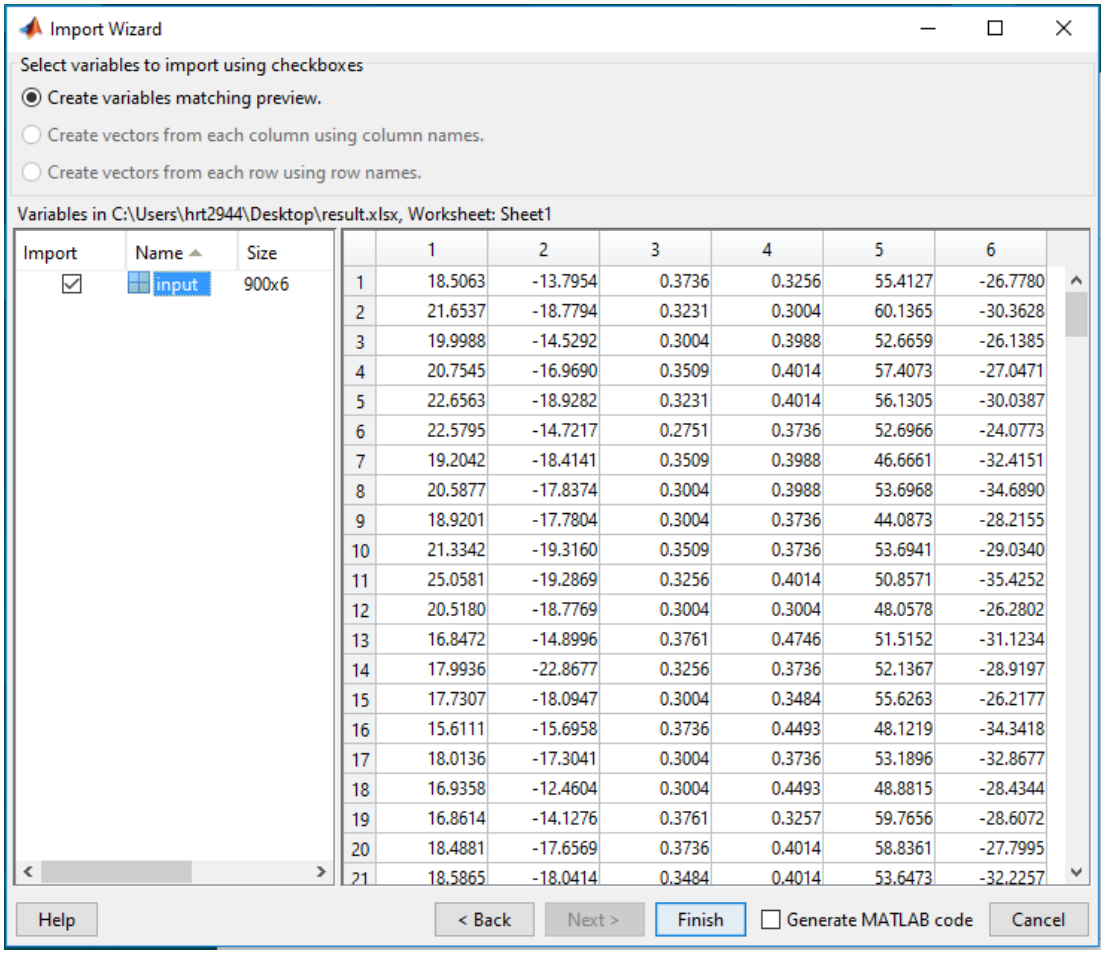



Figure 6.8: Input dataset for ANN classification

### 6.3.1 Parameter Optimization for ANN Classification

#### Scaled Conjugate Gradient backpropagation

The average MSE (in 10 steps) in the ANN training using Scaled Conjugate Gradient backpropagation shown in Table 6.5.

Table 6.5: The average MSE using Scaled Conjugate

| Neuron Number | Average MSE |
|---|---|
| 10 | 0.0925 |
| 20 | 0.0761 |
| 30 | 0.0691 |

As shown in Table 6.5, the average MSE decreases with the increase of the neuron number. The performance of classification is shown in the confusion matrix (Figure 6.9 and Figure 6.10).

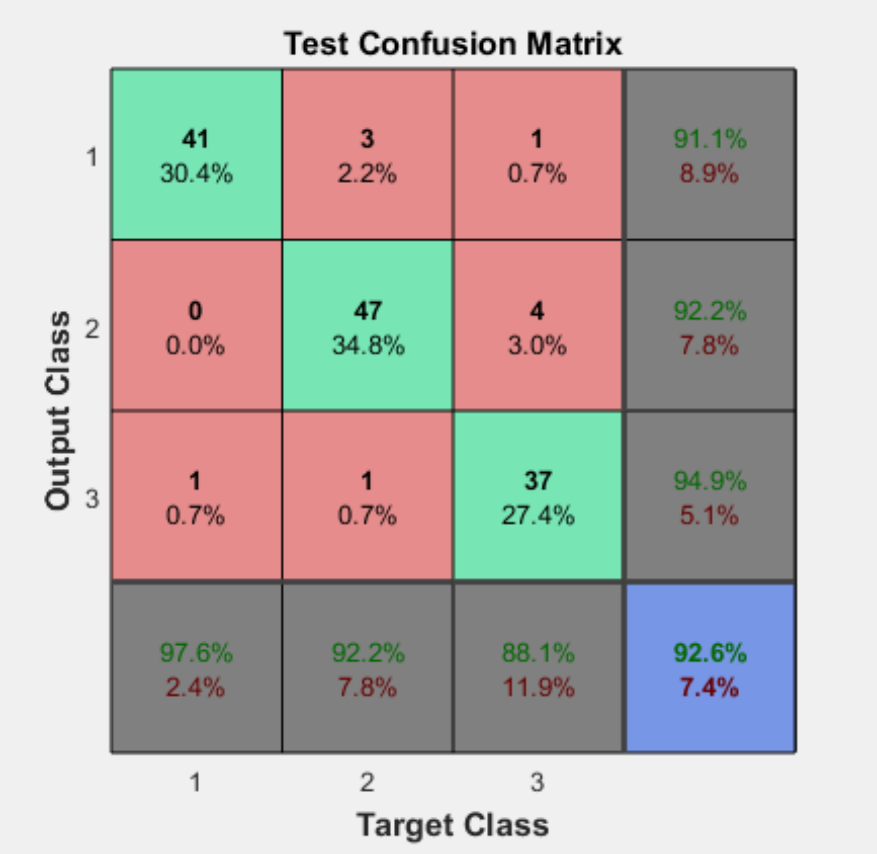



Figure 6.9: Confusion matrix using Scaled Conjugate

The confusion matrix reports the information about the actual and predicted classifications done by a ANN classification (Provost & Kohavi, 1998). In the confusion matrix, there are total 42 samples of Class 1, the system miss predicted that 1 sample to Class 3, and for 51 Class 2 sample, it wrongly predicted that 3 samples to Class 1 and 1 samples to Class 3. As for 42 Class 3 sample, it miss predicted one sample to Class 1 and four samples to Class 2. Overall, the accuracy of Class 1 ,Class 2 and Class 3 were 97.6%, 92.2% and 88.1% respectively. The overall accuracy is 92.6%.

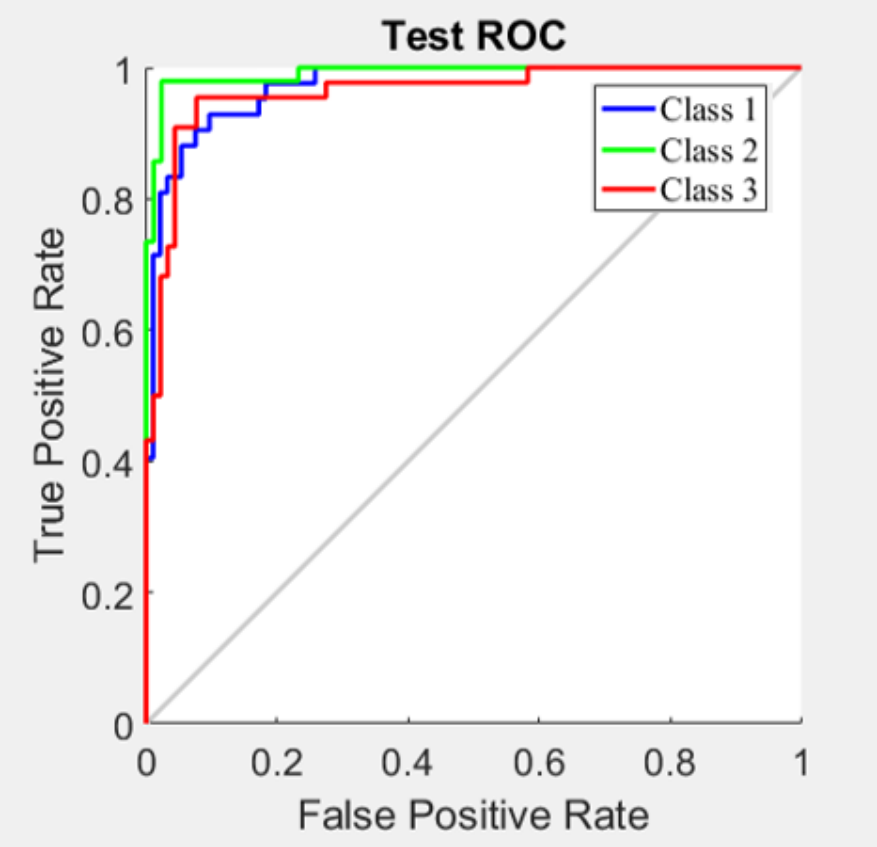



Figure 6.10: Receiver Operating Characteristic using Scaled Conjugate training algorithm

The ROC curve represents the false positive rate on the $X - axis$ and the true positive rate on the $Y - axis$. The more each curve hugs the upper-left corner, the better performance the classification have.

### Gradient Descent backpropagation

The average MSE(in 10 samples) in the ANN training using Levenberg-Marquardt backpropagation is shown in Table 6.6.

Table 6.6: The average MSE using Gradient Descent training algorithm

| Neuron Number | Average MSE |
|---|---|
| 10 | 0.0742 |
| 20 | 0.0767 |
| 30 | 0.0821 |

As shown in Table 6.6, the average MSE changes lightly with the different neuron number. Furthermore, we can demonstrate the performance of classification in each class by using confusion matrix ( Figure 6.11).

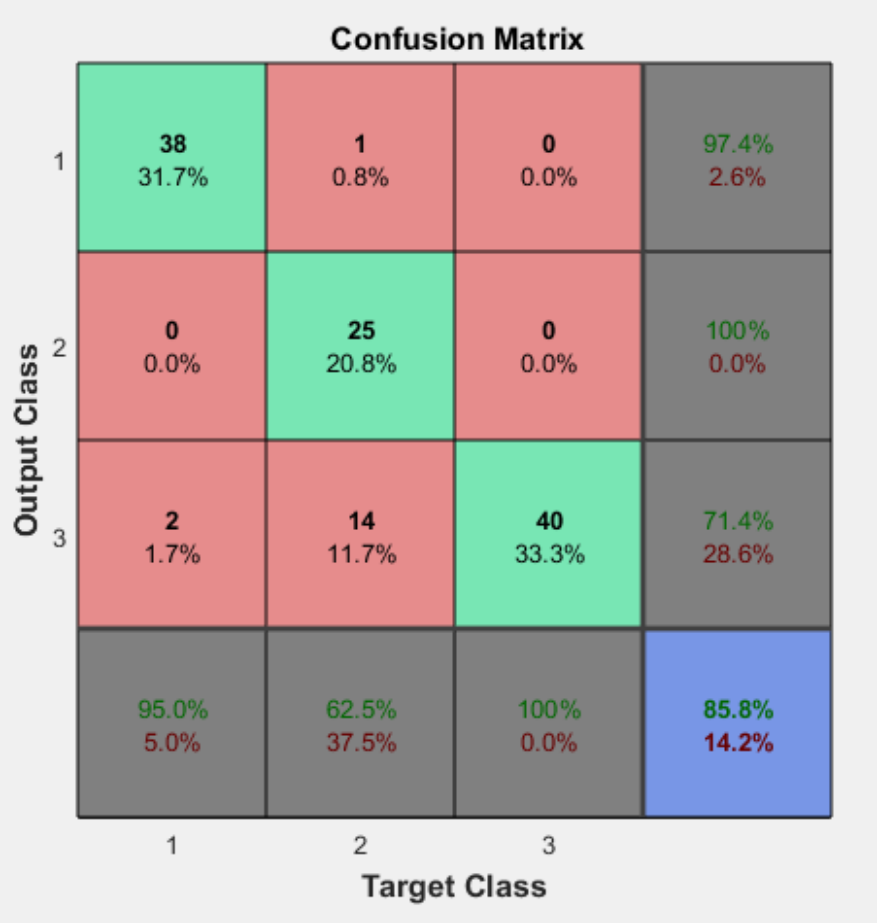



Figure 6.11: Confusion matrix using Gradient Descent

By using Gradient Descent training algorithm, the overall accuracy of classification was 85.8%, which is lower than the accuracy by using Scaled Conjugate training algorithm. The accuracy of Class 2 samples classification was 62.5% in this confusion matrix. For Class 2 samples, the system mis-predicted one sample to Class 1 and 14 samples to Class 3.) while it was 92.2% by using Scaled Conjugate. In contrast, the accuracy of Class 3 sample classification represented 100%, which have a better performance than that by using Scaled Conjugate training algorithm. We can see from the matrix that this algorithm cannot accurately distinguish Class 2 and Class 3.

### Levenberg-Marquardt backpropagation

The average MSE (in 10 samples) in the ANN training using Levenberg-Marquardt backpropagation is shown in Table 6.7.

Table 6.7: The average MSE using Levenberg-Marquardt training algorithm

| Neuron Number | Average MSE |
|---|---|
| 10 | 0.0487 |
| 20 | 0.0519 |
| 30 | 0.0523 |

As shown in Table 6.7, the average MSE increases with the increase of the neuron number. We found that Levenberg-Marquardt backpropagation produces the lowest MSE among three training algorithms. The confusion matrix using Levenberg-Marquardt training algorithm is shown in Figure 6.12

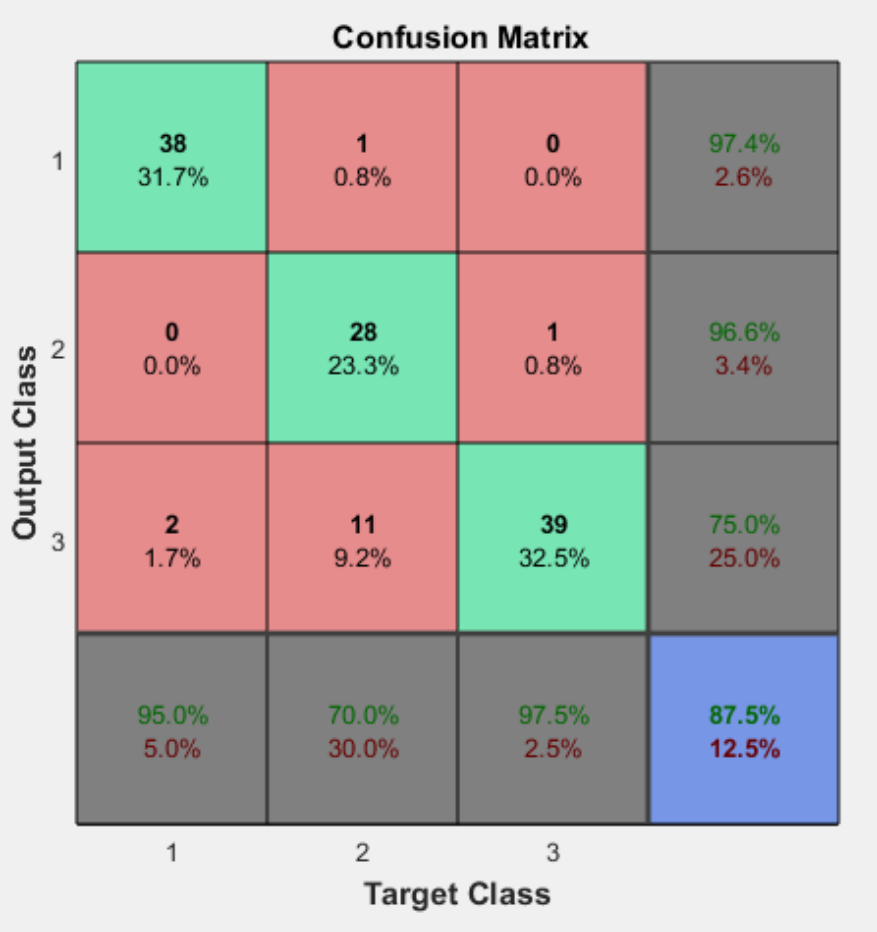



Figure 6.12: Confusion matrix using Levenberg-Marquardt training algorithm

In this confusion matrix, there are total 40 actual Class 1 samples, the system miss-predicted that two samples to Class 3, and for 40 Class 2 samples, it miss-predicted that one sample to Class 1 and 11 samples to Class 3. And for 40 Class 3 samples, it miss-predicted one sample to Class 2. Compared with the other two algorithms, the overall accuracy by using Levenberg-Marquardt training algorithm is lower than using Scaled Conjugate training algorithm but higher than Gradient Descent training algorithm, at 87.5%. The same as using Gradient Descent, Levenberg-Marquardt training algorithm, LM training algorithm also has trouble to distinguishing Class 2 samples from Class 3 samples.

### 6.3.2 The Result of ANN classification

According to previous experiments, we found the best algorithm for ANN is Gradient Descent with 20 neurons. The testing result is shown in Table 6.8.

Table 6.8: ANN classification result

| Trail | Target (m) | ANN output (m) | ANN Mis-classification | Error (m) | Error (%) |
|---|---|---|---|---|---|
| 1 | 12.1 | 12.3 | 1 | 0.3869 | 3.1975 |
| 2 | 12.22 | 12.4 | 1 | 0.349 | 2.8560 |
| 3 | 12.58 | 12.5 | 1 | 0.08 | 0.6359 |
| 4 | 12.58 | 12.5 | 0 | 0.08 | 0.6359 |
| 5 | 12.44 | 12.6 | 1 | -0.16 | -1.2862 |
| Overall | 61.92 | 62.3 | 4 | 0.7359 | 1.1885 |

The result shows that there are four samples are incorrectly classified. These miss-classification caused the error of about $28cm$, which is nearly 40% of the overall error in 50 steps. The ANN classification shows a satisfactory result, the error rate is 1.189%.

## 6.4 Summary

By using the double integration method, the error was significant, though the performance of Simpon's Rule was better than trapezoidal rule. The mean error of estimated stride length was about $30cm$ for each step.

By using the ANN fitting function in MATLAB neural network toolbox, we found that different training algorithm with different neurons can significantly affect performances. According to these experiments, the optimal parameters for ANN fitting process is Levenberg-Marquardt training algorithm with 30 neurons. We tested this network with 50 steps, the total distance was about $61.92m$ and the estimated length was $60.35m$, the error was $1.57m$ which is equivalent to $3.1cm$ for each step, a drastic improvement from double integration method

For ANN classification function, the overall classification accuracy of Scaled Conjugate training algorithm was $92.6\%$ and the overall classification accuracy of Gradient Descent training algorithm was $85.5\%$. , Then, Levenberg-Marquardt training algorithm also has trouble to distinguishing Class 2 as the accuracy of Class 2 sample classification was only $70.0\%$ while the overall accuracy was $87.5\%$. According to these experiments, we used the same testing data set used in ANN fitting to test the performance of the ANN classification function. The result showed that in the total distance of $61.62m$, the neural network estimated that the distance was $62.3m$, the error rate was only $1.1885\%$ which is equivalent to an error of $1.9cm$ for a step

Compared the above results, we found that the ANN classification function had a better performance than the ANN fitting function. The results of these two ANN estimation methods are satisfactory, the error rate is $2.5339\%$ for the ANN fitting function and $1.1885\%$ for the ANN classification function.

# Chapter 7

# Conclusion and Future Work

In this thesis, we demonstrated the ability of using ANN to estimate the step length with low error and variance. Our method has the ability to handle the acceleration and Euler angle data with noise from the low-cost IMU. The experiment result indicated that at a distance of $62.3m$, the error was only $1.1885\%$ by using ANN classification and $2.5339\%$ by using ANN fitting. Compared with traditional double integration methods, due to the step length estimation using trained ANN do not require zero velocity detection and integration. Our method only require a trained ANN to estimate the step length, which simplified the computational procedure.

Due to the limitation of time, we do not implement the method for orientation detection. Moreover, in the ANN classification, we only covered the step length from 115cm to 145cm. In the future work, we will integrate the orientation detection to our method as well as broaden the range of the ANN classification. In this way, our neural network model has the potential to be used as a completed navigation system.

# References


Aggarwal, K., Singh, Y., Ch, P. & Puri, M. (2005). Bayesian regularization in a neural network model to estimate lines of code using function points.

Ahmad, M., Saeed, M., Saleem, S. & Kamboh, A. M. (2016). Seizure detection using eeg: A survey of different techniques. In *Emerging technologies (icet), 2016 international conference on* (pp. 1–6).

Ahn, J. & Han, R. (2012). An indoor augmented-reality evacuation system for the smartphone using personalized pedometry. *Human-Centric Computing and Information Sciences*, *2*(1), 18.

Alvarez, J. C., Alvarez, D., López, A. & González, R. C. (2012). Pedestrian navigation based on a waist-worn inertial sensor. *Sensors*, *12*(8), 10536–10549.

Anwary, A. R., Yu, H. & Vassallo, M. (2017). Optimal foot location for placing wearable imu sensors and automatic feature extraction for gait analysis. *IEEE Sensors Journal*, *18*(6), 2555–2567.

Azami, H., Mohammadi, K. & Bozorgtabar, B. (2012). An improved signal segmentation using moving average and savitzky-golay filter. *Journal of Signal and Information Processing*, *3*(01), 39.

Bachschmidt, R. A., Harris, G. F. & Simoneau, G. G. (2001). Walker-assisted gait in rehabilitation: a study of biomechanics and instrumentation. *Ieee Transactions on neural systems and Rehabilitation Engineering*, *9*(1), 96–105.

Bahl, P. & Padmanabhan, V. N. (2000). Radar: An in-building rf-based user location and tracking system. In *Infocom 2000. nineteenth annual joint conference of the ieee computer and communications societies. proceedings. ieee* (Vol. 2, pp. 775–784).

Bamberg, S. J. M., Benbasat, A. Y., Scarborough, D. M., Krebs, D. E. & Paradiso, J. A. (2008). Gait analysis using a shoe-integrated wireless sensor system. *IEEE transactions on information technology in biomedicine*, *12*(4), 413–423.

Baxt, W. G. (1991). Use of an artificial neural network for the diagnosis of myocardial infarction. *Annals of internal medicine*, *115*(11), 843–848.

Buscema, M. (1998). Back propagation neural networks. *Substance use & misuse*, *33*(2), 233–270.

Cai, G., Chen, B. M. & Lee, T. H. (2011). Coordinate systems and transformations. In *Unmanned rotorcraft systems* (pp. 23–34). Springer.

Castaneda, N. & Lamy-Perbal, S. (2010). An improved shoe-mounted inertial navigation system. In *Indoor positioning and indoor navigation (ipin), 2010 international*

*conference on* (pp. 1–6).

Chen, R., Pei, L. & Chen, Y. (2011). A smart phone based pdr solution for indoor navigation. In *Proceedings of the 24th international technical meeting of the satellite division of the institute of navigation* (pp. 1404–1408).

Chen, Y. & Kobayashi, H. (2002). Signal strength based indoor geolocation. In *Communications, 2002. icc 2002. ieee international conference on* (Vol. 1, pp. 436–439).

Chiang, K.-W., Chang, H.-W., Li, C.-Y. & Huang, Y.-W. (2009). An artificial neural network embedded position and orientation determination algorithm for low cost mems ins/gps integrated sensors. *Sensors*, *9*(4), 2586–2610.

Cho, S. Y. & Park, C. G. (2006). Mems based pedestrian navigation system. *The Journal of Navigation*, *59*(1), 135–153.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, *74*(368), 829–836.

Cleveland, W. S. & Loader, C. (1996). Smoothing by local regression: Principles and methods. In *Statistical theory and computational aspects of smoothing* (pp. 10–49). Springer.

Colwell, S. (2007). Business outlook-the price is right! gps prices drop for consumers. *GPS World*, *18*(5), 32.

Cruz, E. D., Alouani, A. T., Rice, T. R. & Blair, W. D. (1992). Sensor registration in multisensor systems. In *Signal and data processing of small targets 1992* (Vol. 1698, pp. 382–394).

Demuth, H., Beale, M. & Hagan, M. (2008). Neural network toolbox™ 6. *User's guide*, *10*, 11.

Everett, T. & Kell, C. (2010). *Human movement: an introductory text*. Elsevier health sciences.

Fang, L., Antsaklis, P. J., Montestruque, L. A., McMickell, M. B., Lemmon, M., Sun, Y., ... others (2005). Design of a wireless assisted pedestrian dead reckoning system-the navmote experience. *IEEE transactions on Instrumentation and Measurement*, *54*(6), 2342–2358.

Faulkner, W. T., Alwood, R., Taylor, D. W. & Bohlin, J. (2010). Gps-denied pedestrian tracking in indoor environments using an imu and magnetic compass. In *Proceedings of the 2010 international technical meeting of the institute of navigation* (pp. 198–204).

Foxlin, E. (2005). Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Computer graphics and applications*, *25*(6), 38–46.

Gardner, M. W. & Dorling, S. (1998). Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, *32*(14-15), 2627–2636.

Gerstner, W. & Naud, R. (2009). How good are neuron models? *Science*, *326*(5951), 379–380.

Godha, S., Lachapelle, G. & Cannon, M. E. (2006). Integrated gps/ins system for pedestrian navigation in a signal degraded environment. In *Ion gnss* (Vol. 2006).

Grewal, M. S., Weill, L. R. & Andrews, A. P. (2007). *Global positioning systems, inertial navigation, and integration*. John Wiley & Sons.

Gusenbauer, D., Isert, C. & Krösche, J. (2010). Self-contained indoor positioning on off-the-shelf mobile devices. In *Indoor positioning and indoor navigation (ipin), 2010 international conference on* (pp. 1–9).

Hagan, M. T., Demuth, H. B., Beale, M. H. & De Jesús, O. (1996). *Neural network design* (Vol. 20). Pws Pub. Boston.

Hamilton, W. R. (1866). *Elements of quaternions*. Longmans, Green, & Company.

He, S., Lau, R. W., Liu, W., Huang, Z. & Yang, Q. (2015). Supercnn: A superpixelwise convolutional neural network for salient object detection. *International journal of computer vision*, *115*(3), 330–344.

Jain, A. K., Mao, J. & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, *29*(3), 31–44.

Jimenez, A. R., Seco, F., Prieto, C. & Guevara, J. (2009). A comparison of pedestrian dead-reckoning algorithms using a low-cost mems imu. In *Intelligent signal processing, 2009. wisp 2009. ieee international symposium on* (pp. 37–42).

Jin, W., Li, Z. J., Wei, L. S. & Zhen, H. (2000). The improvements of bp neural network learning algorithm. In *Signal processing proceedings, 2000. wccc-icsp 2000. 5th international conference on* (Vol. 3, pp. 1647–1649).

Kao, W.-W., Chen, C.-K. & Lin, J.-S. (2001). Step-length estimation using wrist-worn accelerometer and gps. In *Proceedings of the 24th international technical meeting of the satellite division of the institute of navigation (ion gnss 2011)* (p. 3274).

Khan, J., Wei, J. S., Ringner, M., Saal, L. H., Ladanyi, M., Westermann, F., ... others (2001). Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature medicine*, *7*(6), 673.

Kim, J. W., Jang, H. J., Hwang, D.-H. & Park, C. (2004). A step, stride and heading determination for the pedestrian navigation system. *Positioning*, *1*(08), 0.

Kong, X. (2004). Ins algorithm using quaternion model for low cost imu. *Robotics and Autonomous Systems*, *46*(4), 221–246.

Levi, R. W. & Judd, T. (1996, December 10). *Dead reckoning navigational system using accelerometer to measure foot impacts.* Google Patents. (US Patent 5,583,776)

Li, F., Zhao, C., Ding, G., Gong, J., Liu, C. & Zhao, F. (2012). A reliable and accurate indoor localization method using phone inertial sensors. In *Proceedings of the 2012 acm conference on ubiquitous computing* (pp. 421–430).

Liu, Y., Dashti, M. & Zhang, J. (2013). Indoor localization on mobile phone platforms using embedded inertial sensors. In *Positioning navigation and communication (wpnc), 2013 10th workshop on* (pp. 1–5).

Macdermid, P. W., Fink, P. W. & Stannard, S. R. (2015). Shock attenuation, spatio-temporal and physiological parameter comparisons between land treadmill and water treadmill running. *Journal of Sport and Health Science*.

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.

Meireles, M. R., Almeida, P. E. & Simões, M. G. (2003). A comprehensive review for industrial applicability of artificial neural networks. *IEEE transactions on*

*industrial electronics*, *50*(3), 585–601.

Møller, M. F. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, *6*(4), 525–533.

Motley, A. & Keenan, J. (1988). Personal communication radio coverage in buildings at 900 mhz and 1700 mhz. *Electronics Letters*, *24*(12), 763–764.

Mourcou, Q., Fleury, A., Franco, C., Klopcic, F. & Vuillerme, N. (2015). Performance evaluation of smartphone inertial sensors measurement for range of motion. *Sensors*, *15*(9), 23168–23187.

Mulloni, A., Seichter, H. & Schmalstieg, D. (2011). Handheld augmented reality indoor navigation with activity-based instructions. In *Proceedings of the 13th international conference on human computer interaction with mobile devices and services* (pp. 211–220).

Murata, N., Yoshizawa, S. & Amari, S.-i. (1994). Network information criterion-determining the number of hidden units for an artificial neural network model. *IEEE Transactions on Neural Networks*, *5*(6), 865–872.

Nurunnabi, A., West, G. & Belton, D. (2016). Robust locally weighted regression techniques for ground surface points filtering in mobile laser scanning three dimensional point cloud data. *IEEE Transactions on Geoscience and Remote Sensing*, *54*(4), 2181–2193.

Ojeda, L. & Borenstein, J. (2007). Non-gps navigation with the personal dead-reckoning system. In *Unmanned systems technology ix* (Vol. 6561, p. 65610C).

Panchal, G., Ganatra, A., Kosta, Y. & Panchal, D. (2011). Behaviour analysis of multilayer perceptronswith multiple hidden neurons and hidden layers. *International Journal of Computer Theory and Engineering*, *3*(2), 332.

Pannese, E. (2015). I. neurons and interneuronal connections: A historical overview. In *Neurocytology: Fine structure of neurons, nerve processes, and neuroglial cells* (pp. 1–7). Cham: Springer International Publishing. Retrieved from `https://doi.org/10.1007/978-3-319-06856-5_1` doi: 10.1007/978-3-319-06856-5_1

Payeur, P., Le-Huy, H. & Gosselin, C. M. (1995). Trajectory prediction for moving objects using artificial neural networks. *IEEE Transactions on Industrial Electronics*, *42*(2), 147–158.

Pohtongkam, S. & Srinonchat, J. (2016). Object recognition from human tactile image using artificial neural network. In *Electrical engineering/electronics, computer, telecommunications and information technology (ecti-con), 2016 13th international conference on* (pp. 1–6).

Provost, F. & Kohavi, R. (1998). Guest editors' introduction: On applied research in machine learning. *Machine learning*, *30*(2), 127–132.

Qian, J., Ma, J., Ying, R., Liu, P. & Pei, L. (2013). An improved indoor localization method using smartphone inertial sensors. In *Indoor positioning and indoor navigation (ipin), 2013 international conference on* (pp. 1–7).

Rampp, A., Barth, J., Schülein, S., Gaßmann, K.-G., Klucken, J. & Eskofier, B. M. (2015). Inertial sensor-based stride parameter calculation from gait sequences

in geriatric patients. *IEEE transactions on biomedical engineering*, *62*(4), 1089–1097.

Roweis, S. (1996). Levenberg-marquardt optimization. *Notes, University Of Toronto*.

Sato, H. (2001, October 16). *Moving average filter.* Google Patents. (US Patent 6,304,133)

Savitzky, A. & Golay, M. J. (1964). Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, *36*(8), 1627–1639.

Schafer, R. W. (2011). What is a savitzky-golay filter?[lecture notes]. *IEEE Signal processing magazine*, *28*(4), 111–117.

Shatz, C. J. (1992). The developing brain. *Scientific American*, *267*(3), 60–67.

Shin, S., Park, C., Kim, J., Hong, H. & Lee, J. (2007). Adaptive step length estimation algorithm using low-cost mems inertial sensors. In *Sensors applications symposium, 2007. sas'07. ieee* (pp. 1–5).

Shoeb, A. H. (2009). *Application of machine learning to epileptic seizure onset detection and treatment* (Unpublished doctoral dissertation). Massachusetts Institute of Technology.

Shultz, S. P., D'hondt, E., Fink, P. W., Lenoir, M. & Hills, A. P. (2014). The effects of pediatric obesity on dynamic joint malalignment during gait. *Clinical biomechanics*, *29*(7), 835–838.

Stirling, R., Collin, J., Fyfe, K. & Lachapelle, G. (2003). An innovative shoe-mounted pedestrian navigation system. In *proceedings of european navigation conference gnss* (Vol. 110).

Stirling, R. G. (2004). *Development of a pedestrian navigation system using shoe mounted sensors* (Unpublished doctoral dissertation). University of Alberta.

Stone, E. E. & Skubic, M. (2011). Passive in-home measurement of stride-to-stride gait variability comparing vision and kinect sensing. In *Engineering in medicine and biology society, embc, 2011 annual international conference of the ieee* (pp. 6491–6494).

Svozil, D., Kvasnicka, V. & Pospichal, J. (1997). Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, *39*(1), 43–62.

Terrier, P., Ladetto, Q., Merminod, B. & Schutz, Y. (2000). High-precision satellite positioning system as a new tool to study the biomechanics of human locomotion. *Journal of Biomechanics*, *33*(12), 1717–1722.

Tetko, I. V., Livingstone, D. J. & Luik, A. I. (1995). Neural network studies. 1. comparison of overfitting and overtraining. *Journal of chemical information and computer sciences*, *35*(5), 826–833.

Trojaniello, D., Ravaschio, A., Hausdorff, J. M. & Cereatti, A. (2015). Comparative assessment of different methods for the estimation of gait temporal parameters using a single inertial sensor: application to elderly, post-stroke, parkinson's disease and huntington's disease subjects. *Gait & posture*, *42*(3), 310–316.

Truong, P. H., Lee, J., Kwon, A.-R. & Jeong, G.-M. (2016). Stride counting in human walking and walking distance estimation using insole sensors. *Sensors*, *16*(6), 823.

Weisstein, E. W. (2009). Euler angles.

Willemsen, A. T. M., Bloemhof, F. & Boom, H. B. (1990). Automatic stance-swing phase detection from accelerometer data for peroneal nerve stimulation. *IEEE Transactions on Biomedical Engineering*, *37*(12), 1201–1208.

Woodman, O. & Harle, R. (2008). Pedestrian localisation for indoor environments. In *Proceedings of the 10th international conference on ubiquitous computing* (pp. 114–123).

x-io Technologies Ltd. (2017). Ngimu-user-manual-v1.3.

Ying, H., Silex, C., Schnitzer, A., Leonhardt, S., Schiek, M., Leonhardt, S., ... Magjarevic, R. (2007). 4th international workshop on wearable and implantable body sensor networks. *Springer Berlin Heidelberg*.

Yuste, R. & Denk, W. (1995). Dendritic spines as basic functional units of neuronal integration. *Nature*, *375*(6533), 682.