Development of Equine Gait Recognition Algorithm

Sigrid Björnsdotter & Mikael Maga

Master's Thesis
in Biomedical Engineering

Spring 2017

LUND
UNIVERSITY

Faculty of Engineering
Department of Biomedical Engineering
Supervisor: Frida Sandberg

Sony Mobile Communications AB

SONY

Abstract

Horseback riding is a sport enjoyed by people around the world. Many riders are interested in knowing exactly how much they have exercised their horse and how much time that have been spent in different gaits. The goal of this master's thesis was to develop an equine gait recognition algorithm. Triaxial accelerometer and gyroscope signals were collected during different riding sessions by using smartphones. Features, used in previous activity recognition works, were implemented and calculated for all sensor signals. Different methods to select important features were used and the feature sets were then evaluated. In the work four classifiers were implemented and evaluated.

The work resulted in an equine gait recognition algorithm based on signals collected at the saddle-girth. The developed algorithm used a window length of 128 samples and windows with 50 % overlap. A feature set was chosen by the use of sequential forward feature selection. Five features were included in the final algorithm and two classifiers using two respectively three of the features. The first classifier separated stand from the gaits by using the features root mean square for the magnitude of the gyroscope signal and energy of the x-axis accelerometer signal. The second classifier classified gaits as either walk, trot or canter using the wavelet based feature energy distribution ratio of the z-axis accelerometer signal, dominant frequency of z-axis of the gyroscope signal and skewness of the accelerometer z-axis. The classifiers used in both classification steps were KNN with K = 3.

The algorithm performed well on a collected test set including two riding sessions. It should be noted that the same phone was used to collect both training and testing data. The performance of the developed algorithm was benchmarked against the smartphone application Equilab. The performance of both algorithms was similar. The developed equine gait recognition algorithm had a 94.1 % and 97.4 % accuracy on the two different test sessions.

Further development of the algorithm will be needed to include other terrains and a larger variety of horses and riders.

Keywords: Activity Recognition - Gait Analysis - Horse Back Riding - Accelerometer - Gyroscope

Preface

This work is a thesis for the degree Master of Science in Biomedical Engineering at the Faculty of Engineering, Lund University (LTH). The project was carried out at Sony Mobile Communications AB in Lund. Huge thanks to our supervisors Frida Sandberg at LTH and Andrej Petef at Sony Mobile Communications for all your support. We also would like to thank Aminda Ingulfson and Hanna Sassner and all riders and horses at Flyinge, without your help with data collection this work would not have been possible. Finally, we would like to thank all other people at Sony Mobile Communications who have helped us in different ways.

Contents

1. Introduction

1.1 Introduction to the Master's Thesis

In a world of Internet of Things (IoT) devices, with sensors embedded in almost any electronic device, sensors are also attached to humans and animals. By monitoring motion patterns, means for understanding movement and behaviour in different situations will be provided. Many IoT related applications can be foreseen, such as monitoring human and animal activities, guiding athletes to improved motion patterns, or possibly indicating health problems early on. One of the possible areas for applications is horseback riding.

Horseback riding is a sport enjoyed by people around the world. The horse is an active animal by nature and when horses still lived as wild horses they moved many hours every day to find food. Horses kept in stables thereby needs a lot of exercise every day [1]. Exercise is important for the functioning of the blood circulation and the lymphatic fluid circulation of the horse. Horses do not move without a reason and need to be exercised or put out to pasture [2].

The horse moves in many different gaits, the common natural gaits are walk, trot and canter. The walk is a 4-beat movement that happens in eight steps where the horse alternates between three respective two legs on the ground. The trot is a 2-beat movement and the movement happens in four steps. The diagonal leg pairs work together, and should touch the ground on the same time. The canter is a 3-beat movement that happens in six steps, it alternates between one leg, three legs, two legs, three legs and one leg on the ground followed by suspension [1].

Many riders are interested in knowing exactly how much they have exercised their horse and how much time that have been spent in different gaits. Riders' experience of the riding session does not always match what has been achieved during riding session. An objective analysis of gaits could help the riders customize the training to reach their desired goals. It could also make it easier to keep track of the training of a horse if many riders are involved. A smartphone application used for gait recognition, Equilab, was selected the product of the year 2016 by the readers of the horse magazine Hippson [3]. However, riders we talked to complained that the application had gait recognition problems and many misclassifications were done.

The goal of this master's thesis was thereby to develop an algorithm for equine gait recognition, collect a unique data set for the development of the algorithm and benchmark the developed algorithm against the existing product Equilab. The algorithm should be able to classify the gaits, walk, trot and canter, as well as stand. An additional goal was to create a useful Matlab script for analysis of different kinds of motion patterns. The script should contain a systematic method to identify valuable features for classification and evaluate different classifiers.

Since there are a lot more research done in human activity recognition, the starting point of this project was inspired by human activity recognition methods combined

with knowledge of horse gaits.

This thesis work starts with a survey of the field. It continues with chapter 2 where the sensors used are described followed by time domain, frequency domain and wavelet based feature characterization and a description of the different classifiers used. The chapter ends with different feature selection and feature reduction methods. Chapter 3 describes different choices made for the data collection and how the data was collected. Chapter 4 explains the process of preparing the collected training data for further analysis. In chapter 5 the algorithm development process is described and the final equine gait recognition algorithm is presented. In chapter 6 the collection of testing data and the results of the developed algorithm are presented. The results are also benchmarked against the smartphone application Equilab. In chapter 7 the results of this thesis work and future work are discussed. The thesis work ends with chapter 8 where the most important conclusions are presented.

1.2 Survey of the Field

1.2.1 Related Work

1.2.1.1 Human Activity Recognition

There are several studies done on different motion patterns. Two common ways to study motion patterns are image analysis and use of accelerometer data. In this master's thesis the focus will be on activity recognition using accelerometer and gyroscope data. One of the most studied group of activities, using this kind of sensor data, is basic human everyday movements, such as walking, stair climbing, stair descending, running, sitting and laying down. Recognition of different hand gestures is also a well-studied area using sensor data, and used in commercial products such as the Wii-controller [4]. An increasing number of studies are done on animal movement, for example activity recognition of cattle [5][6]. The methods used for analyzing motion patterns with sensor data differ and below some different studies are presented.

Bao and Intille [7] has made a work on human activity recognition. Their work is one of the most cited works in this field since they proved that accelerometer signals are very useful for human activity recognition. Algorithms were developed and evaluated using data from five biaxial accelerometers worn simultaneously on different parts of the body. The subjects performed 20 different everyday activities (for example walking, watching TV, brushing teeth, climbing stairs, vacuuming) without being told specifically how and where to do them. The features extracted from the data were mean, energy, frequency-domain entropy, and correlation of acceleration data. Using theses features, the classifiers decision table, instance-based learning, C4.5 decision tree, and naive Bayes classifiers were trained and tested. The decision tree generated the highest recognition accuracy, over 80 %. It was also noticed that high accuracy was obtained using only two accelerometers placed on thigh and wrist.

The dataset collected by Bao and Intille [7] was also used by Mannini and Sabatini [8] but reduced to only contain data from seven activities. They give a review on how human physical activity can be classified using on-body accelerometers. Especially, hidden Markov models are discussed but also classifiers such as K-nearest neighbor (KNN), artificial neural networks (ANN) and support vector machine (SVM).

Many different works use the classifier hidden Markov models since it has been shown to be a good classifier for human activity recognition (see for example [4][8][9]). A recent published article by Ronao och Cho [9] propose a classifier architecture of continuous hidden Markov models. A two-stage continuous hidden Markov model framework is used to analyze accelerometer and gyroscope data. The feature selection is done by using random forest variable importance measures (RF VI), a technique combining bagging and random feature selection by creating a collection of simple decision trees.

To be able to recognize many different types of activities, the combination of a wrist-worn sensor and pocket worn sensor has been evaluated by Shoaib et al. [10]. The study included recognition of thirteen activities, divided into two groups, simple and complex. The simple group contained repetitive activities such as walking and jogging. The complex group contained motions such as smoking, eating, drinking coffee and giving a talk. The work discusses activity classification techniques, sensor combinations and different window sizes. Accelerometer, gyroscope and linear accelerator were used. A feature set including mean, standard deviation, minimum and maximum value, median, semi-quartile and the sum of the first ten Fast Fourier Transform (FFT) coefficients was used to be able to classify both simple and complex patterns. For classification three different methods were used, naive Bayes, KNN and decision tree. It was concluded that the recognition performance of complex patterns was improved when data from both the wrist-worn sensor and pocket sensor were used for analysis. Combining both accelerometer and gyroscope data generated better recognition performance than only accelerometer data. The effect of different window sizes was also analyzed. Small window sizes worked good for recognizing simple tasks but larger window sizes were better for classifying more complex tasks.

Another approach for human activity recognition is template matching. Margarito et al. [11] have done activity recognition for eight common sport activities (cycling, cross training, rowing, squatting, stepping running walking, and weightlifting) by using a single triaxial accelerometer placed at the wrist. Template-based activity recognition was done by creating templates based on a training set and then compare example signals and the templates for classification. The comparison was made by using five different similarity measures (Euclidean distance, dynamic time warping (DTW), derivative DTW, cross-correlation, and combining distance and correlation metrics (Rce)). It was concluded that Rce generated highest classification accuracy, approximately 80 %. The template-matching-based activity recognition was compared to the more common statistical-learning classifiers naive Bayes (NB), logistic regression (LR), decision tree and artificial neural network (ANN) using 11 time and frequency features (mean, variance, root mean square of the derivative signal, min-max range, total energy, skewness, main frequency, entropy, quantile 0.25, quantile

0.50 and quantile 0.75). ANN and LR generated the highest recognition accuracy, approximately 85 % which is a bit higher than for the best template-based recognition method.

There are also studies where only gyroscope data have been used to classify motion patterns, an example of this is a work done by Barshan and Ayrulu-Erdem [12]. Two single-axis gyroscopes placed on the right leg, above and under the knee, were used to collect data. To classify eight different leg motions an ANN with features extracted using the discrete wavelet transform (DWT) were used. Different wavelet types were tested. The advantage of using wavelet transform compared to Fourier methods is that wavelet transform can perform local analysis and thereby analyze signals that are highly non-stationary, noisy, and aperiodic whereas Fourier retrieves the global frequency content.

As can be seen there are many different approaches when analyzing human motion patterns. Many different techniques for feature extraction and classification have been tested. The sensor placements seem to depend on what kind of activities that are analyzed.

1.2.1.2 Equine Activity Recognition

Analysis of horses using accelerometers has successfully been done by different research groups [13][14][15][16]. They have analyzed, for example, different gaits, jumping techniques, horse-rider coordination and lameness.

Gait determination using accelerometers has been done by for example Burla et al. [13] and Robilliard et al. [14]. Burla et al. used 20 horses of different breeds and heights. An accelerometer was placed on left foreleg and the vertical acceleration was measured with the sampling frequency 10 Hz. The mean of the absolute acceleration value for each of the different gaits stand, walk, trot and gallop was calculated. By taking breed classes into account, the acceleration values of the different gaits did not overlap. The use of accelerometer was therefore considered a good way to measure locomotor activity and resting behavior in horses. Robilliard et al. did characterize different gaits of horses by using the foot-fall (foot-on and foot-off) pattern. Eight Icelandic horses were used in the study and were analyzed by using accelerometers attached to the dorsal hoof of all four limbs. To find an optimal criteria to differentiate gaits the method linear discriminant analysis (LDA) was used. The LDA made it possible to distinguish between symmetric and asymmetric gaits.

Barrey and Galloux [15] did a work analyzing equine jumping technique with accelerometer where the vertical acceleration at the sternum was measured. The measurements were made on eight horses running a course with 14 obstacles. The four best horses during the race were considered “good jumpers” and the other four were considered “poor jumpers”. Both time domain analysis and Morlet’s wavelet analysis were used to analyze jumping technique. From the recorded accelerometer data the peak acceleration was extracted from the forelimbs and hindlimbs just before the take off. It was found that the “poor jumpers” had a lower stride frequency and a

higher forelimb/hindlimb acceleration ratio. The results from Morlet's wavelet analysis showed that a faulty jump (knocked over pole) had a decreasing stride frequency at take-off, had absence or a low energy content in the middle frequency range and the relative energy of the acceleration impulse at take-off was considerably smaller than the landing energy of the forelimbs.

Accelerometers have also been used to analyze the coupling between the horse and the rider. One example is the work by Viry et al. [16] who studied the patterns of horse-rider coordination during endurance race using two triaxial accelerometers placed on the horses and the riders. Gait patterns of the horses were identified by the stride frequency. Identification of riding techniques per horse gait was done by quantifying the amplitude of the craniocaudal displacement of the rider. Identification of horse-rider coordination was done by representing the spatio-temporal relationship between the rider and the horse on Lissajous plots.

1.2.2 Products used for Activity Recognition

Many different products have been used for human activity recognition. Examples are different activity bands used on the wrist or apps. But there are also some products for easily measuring activities and wellness of horses that have been launched on the market. The products often use accelerometers, gyroscopes, magnetometers and GPS.

Equilab [17] is a product used for gait activity recognition and to give feedback on training of the horse. The product is an app that uses the sensors in the phone to measure the movements of the horse while riding. The phone is placed in a pocket where it lies steady. Gait, stride, beat, pace, distance, energy consumption, performance and trends are analyzed. Equilab was, as mentioned earlier, selected the product of the year 2016 by the readers of the horse magazine Hippson. The developed algorithm in this thesis is benchmarked against Equilab [18].

Equisense [19] is a French startup company that has developed two different products for horses, Equisense Motion and Equisense Care. Equisense Motion is a sensor used to provide feedback of the rider's training session and the fitness of the horse. The sensor is placed on the saddle-girth and is a 9-axis inertia motion sensor (3-axis gyroscope, 3-axis accelerometer, 3-axis magnetometer) connected to an app. In the app symmetry, time spent at each lead and each gait, elevation, cadence, number of jumps and work intensity can be studied. Equisense Care will be delivered in September 2017. It is a product for measuring the horse well-being and health. The product offers continuous information of for example heart rate, respiratory rate and perspiration, possibility to evaluate the horse's stress level and to follow the daily activity. This is done by using an app connected to a sensor placed on the breast in a bodysuit. The sensor consists of a 6-axis motion sensor (3-axis accelerometer and 3-axis gyroscope), ultra-wideband radar, infrared thermometer and humidity sensor.

Another product very similar to Equisense Care is Nightwatch [20]. Nightwatch is a product used for monitoring equine distress and wellness. The product offers real-time monitoring and alert notifications 24 hours a day. The product monitors

heart rate and respiratory rate by using ultra-wideband impulse radar and monitors activity, motion and posture by using accelerometers, gyroscopes, compasses and altimeters. The sensors are embedded in either a halter or a collar.

Swedish University of Agricultural Sciences (SLU) is performing research on lameness for equines by using a wireless inertial sensor system called Lameness Locator. The sensor system consists of two accelerometers and a gyroscope and is used for data collection and analysis. In an article by McCracken et al. [21] the Lameness Locator was tested in a lameness study. The results showed that the system was able to identify the limb with induced lameness before three equine veterinarians agreed on the affected limb. In the study three sensors were used to measure vertical movement. The sensors were placed on the head, pelvic and right front limb. However, it should be noted that some of the authors have personal interest in the Lameness Locator (co-inventor, seller and stakeholder).

2. Background

In this chapter, sensors, features, classifiers, feature selection and feature reduction methods used in this work are descried.

2.1 Sensors used in Activity Recognition

The most common sensor used in activity recognition analysis is the accelerometer but the gyroscope has also been used. Studies have shown that the two sensors can be used in combination for improved activity recognition [22]. Triaxial accelerometers and gyroscopes can be found in almost all smartphones today.

2.1.1 Accelerometer

An accelerometer measures the acceleration forces in the unit m/s^2. An accelerometer consists of two fundamental parts, a housing and a mass. The housing is attached to the object whose acceleration is measured. The mass is tethered to the housing but can still move. In smartphones the accelerometer is very small and microelectromechanical system (MEMS) technology is used. The accelerometer is made of silicon. In smartphones the housing is attached to the phone and the mass has a comb-like structure. The comb-section can move back and fourth in the silicon tethering to the housing. Fingers surround the comb-section and make up a differential capacitor. Current will flow when the comb-section moves and the amount of flowing current can be correlated to the acceleration. The accelerometer with its fundamental parts is illustrated in Figure 2.1. The acceleration is often measured in three dimensions (x, y and z) in smartphones, see Figure 2.2 [23].

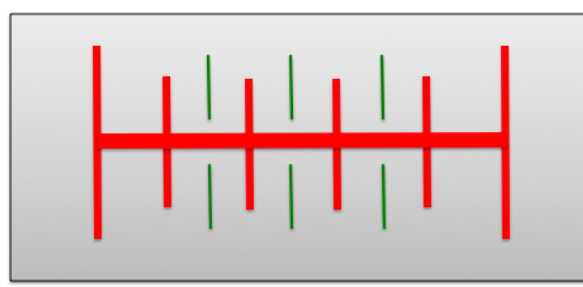

Figure 2.1: Illustration of an accelerometer. The gray box illustrates the housing. The red section is the mass which can move. The green sections illustrates the fingers.

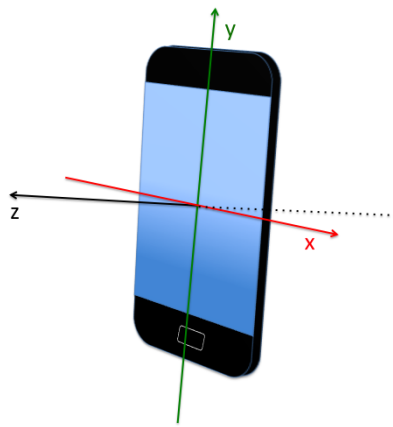

Figure 2.2: Coordinate system of the smartphone.

2.1.2 Gyroscope

The gyroscope is a type of inertial sensor. The gyroscope measures the rate of rotations and the unit is radians per second (rad/s). Gyroscopes in smartphones often uses MEMS technology which contains a vibrating mass. It utilizes the Coriolis effect. The mass is loosely coupled to a housing. When the housing starts to rotate the vibrating mass does not want to change its direction, causing a displacement compared to the housing. This displacement to the housing can be measured in the same way as the accelerometer, using a comb-like structure surrounded by fingers, but the current flow is correlated to a rotational change. The gyroscope also often measure along three dimensions (x, y and z) in smartphones [24].

2.2 Feature Characterization

Features used in successful activity recognition have been calculated in the time domain, the frequency domain and by using wavelet analysis. A mathematical background for better understanding of different features is presented in the following sections. In these sections, $x(n)$ is a time series signal (accelerometer or gyroscope signal) where n refers to the sample. N is the number of samples in a window of the signal.

2.2.1 Time Domain

Mean

The mean value is defined as

$$\mu_x = \frac{1}{N}\sum_{n=1}^{N} x(n) \tag{2.1}$$

Median
The median is the center value when values are sorted in increasing order. If the signal contains an even number of values, the median is the mean of the two center values.

Quartile
Quartile is a statistical term used when a data set is divided into four equal parts. A data set is sorted with the lowest number first and the highest number last. The first quartile (Q1) is the value that splits off the lowest 25% of the data from the highest 75%. The second quartile (Q2) is the value that splits the data in half (the median). The third quartile is the value that splits of the highest 25% of the data from the lowest 75%. The interquartile range (IQR) is the difference between Q3 and Q1. The different quartiles are illustrated in Figure 2.3.

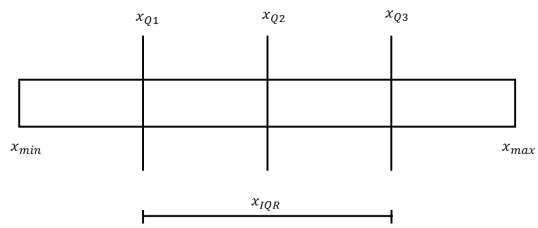

Figure 2.3: Illustration of the quartiles ($Q1$, $Q2$ and $Q3$) and the interquartile range (IQR) of signal x.

Variance
The variance is defined by

$$\sigma_x^2 = \frac{1}{N-1}\sum_{n=1}^{N}(x(n) - \mu_x)^2 \tag{2.2}$$

Standard Deviation
The standard deviation is defined by

$$\sigma_x = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}(x(n) - \mu_x)^2} \tag{2.3}$$

Skewness
Skewness is a measure related to the shape of the probability distribution, it describes the degree of asymmetry in the data. The skewness is negative if most of the mass of the distribution is to the right and the left tail is longer than the right tail. The skewness is positive if most of the mass is to the left and the right tail is longer. If the distribution is symmetric the skewness is 0. Negative and positive skew are

illustrate in Figure 2.4.

$$skew = \frac{1}{N\sigma_x^3}\sum_{n=1}^{N}(x(n) - \mu_x)^3 \tag{2.4}$$

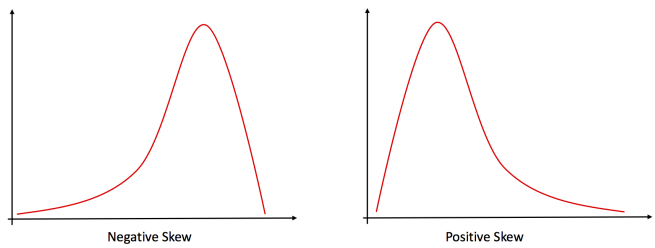

Figure 2.4: Illustration of negative and postive skew.

Kurtosis
Kurtosis is also a measure related to the shape of the probability distribution, it describes if the data is light-tailed or heavy-tailed. Low kurtosis indicates light tails and no outliers and high kurtosis indicates heavy tails or outliers. Normal distribution of data equals kurtosis 3. Low and high kurtosis are described in Figure 2.5.

$$kurt = \frac{1}{N\sigma_x^4}\sum_{n=1}^{N}(x(n) - \mu_x)^4 \tag{2.5}$$

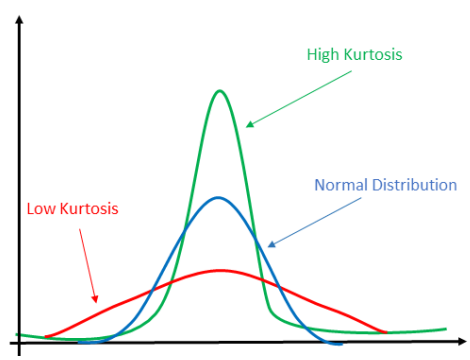

Figure 2.5: Illustration of low and high kurtosis.

Root Mean Square
Root mean square is the square root of the mean squared.

$$x_{rms} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} x(n)^2} \tag{2.6}$$

Crest Factor
The crest factor describes the peaks in a waveform and is a measurement of how extreme the peaks are. If there are no peaks in the signal the crest factor is equal to 1. For a sine wave the crest factor is $\sqrt{2}$.

$$C = \frac{|x|_{peak}}{x_{rms}} \tag{2.7}$$

where $|x|_{peak}$ is defined as half the maximum to minimum range (see below).

Energy
The energy of a discrete time signal, $x(n)$, is defined as

$$E = \sum_{n=1}^{N} |x(n)|^2 \tag{2.8}$$

Range (Maximum - Minimum)
The range is the difference between the maximum value and the minimum value of a signal. The range is described in Figure 2.6.

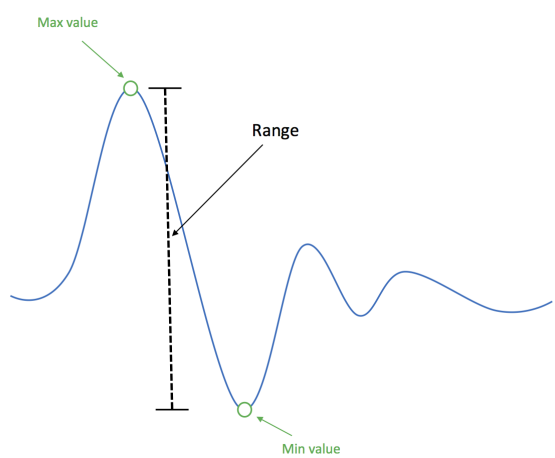

Figure 2.6: Illustration of the maximum - minimum range.

Mean-Crossings
Mean-crossings is a measure of the number of times a signal crosses the mean value, or actually the mean value plus a certain threshold. The threshold in this work was set to ±20 % of the mean value. The measure is illustrated in Figure 2.7.

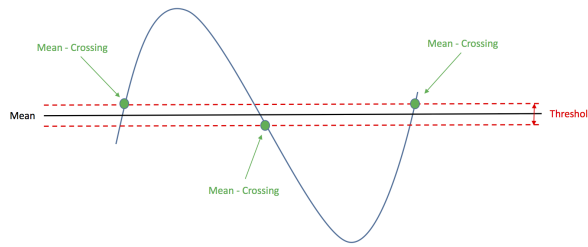

Figure 2.7: Illustration of the feature mean-crossings.

Maximum of Sample Differences
The maximum of sample differences is the maximum sample difference between corresponding samples for two signals, $x(n)$ and $y(n)$.

$$max\ diff = max(x(n) - y(n)), \qquad n = 1, ..., N \tag{2.9}$$

Cross-Correlation
Cross-correlation is a measure of the similarity between two time series. The coefficients are calculated by shifting one of the signals in time with respect to the other signal. If the first signal x is N points and the second signal y is M points, cross-correlation is calculated in the following way

$$\hat{R}_{yx}(\tau) = \frac{1}{N-1} \sum_{n=0}^{N-1} y(n+\tau)x(n) \qquad \tau = 0, ..., M-1 \tag{2.10}$$

Correlation Coefficient
The most commonly used correlation coefficient is the Pearson's product moment coefficient. It is calculated in the following way

$$\rho_{x,y} = \frac{cov(x,y)}{\sigma_x \sigma_y}, \qquad |\rho_{x,y}| \leq 1 \tag{2.11}$$

where the covariance, $cov(x, y)$, is calculated as

$$cov(x,y) = \frac{1}{N-1} \sum_{n=1}^{N} (x(n) - \mu_x)(y(n) - \mu_y) \tag{2.12}$$

2.2.2 Frequency Domain

The discrete Fourier transform is calculated using the fast Fourier transform. The discrete Fourier transform is defined as

$$Y(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-2\pi i}{N} nk} \qquad k = 0, 1, 2, ..., N-1 \tag{2.13}$$

Sum Largest Fourier Coefficients
The coefficients in the discrete Fourier transform is defined by

$$c(k) = \frac{|Y(k)|}{N} \qquad k = 0, 1, 2, ..., N-1 \tag{2.14}$$

One feature is the sum of the m largest Fourier coefficients, where m in this work varies from 1 to 10.

Dominant Frequency
The dominant frequency is the frequency corresponding to the highest peak of the frequency spectrum, i.e. the largest Fourier coefficient (excluding the DC-component). The dominant frequency is illustrated in Figure 2.8.

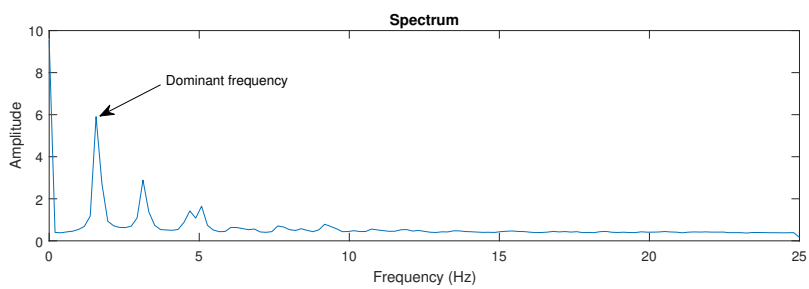

Figure 2.8: The dominant frequency in the spectrum.

Frequency Domain Entropy
The equation for Shannon entropy is given by

$$H(x) = -\sum_{x} p(x) log_2 p(x) \tag{2.15}$$

where $p(x)$ is the probability distribution. In the frequency domain $p(x)$ is given by

$$p(x) = \frac{c(k)}{\sum_{j=1}^{n} c(j)} \qquad k = 0, 1, 2, ..., N-1 \tag{2.16}$$

Frequency Domain Quartile
The same method as used for calculating the time domain quartiles is used when calculating the frequency domain quartiles. The Fourier coefficients are sorted with

the smallest coefficient first and the largest last. The first quartile (Q1) is the value that splits off the lowest 25% of the Fourier coefficients from the highest 75%. The second quartile (Q2) is the median value and the third quartile is the value that splits off the highest 25% of the Fourier coefficients from the lowest 75%.

2.2.3 Wavelet Analysis (Discrete Wavelet Transform)

In wavelet transform certain wavelets such as Haar and Daubechies are used for analysis. The transform is a time-frequency transformation i.e. both information about frequency and location in time are obtained. In the Discrete Wavelet Transform (DWT) a wavelet function and a scaling function are used. The DWT uses the wavelet and its scaling function to construct filter pairs. The filter constructed from a wavelet function works like a high-pass filter and from the complementary scaling function a low-pass filter is constructed.

By convolving the signal $x(n)$ with the low-pass filter, $g(n)$, and a high-pass filter, $h(n)$, two new vectors are acquired, by decimating by two the approximation coefficients $cA_i(n)$ (low-pass), and detail coefficients $cD_i(n)$ (high-pass) are acquired. Where i denotes the level, i.e., $cA_i(n)$ = approximation coefficients for level i, $cD_i(n)$ = Detail coefficients for level i and $cA_0(n) = x(n)$, see Equation 2.17 and 2.18 [25]. In Figure 2.9 the procedure is illustrated.

$$cA_i(n) = \sum_{k=-\infty}^{\infty} cA_{i-1}(k)g(2n-k) \tag{2.17}$$

$$cD_i(n) = \sum_{k=-\infty}^{\infty} cA_{i-1}(k)h(2n-k) \tag{2.18}$$

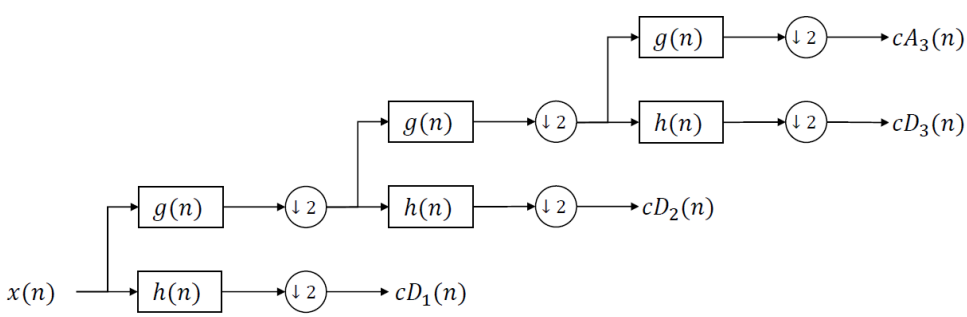

Figure 2.9: An illustration of how the discrete wavelet transform calculates the different coefficients.

Total Energy DWT decomposition
The total energy (E_T) at level i of the DWT decomposition is given by

$$E_T = cA_i cA_i^T + \sum_{j=1}^{i} cD_j cD_j^T \tag{2.19}$$

where

$$cA_i = [cA_i(1)...cAi(N)]$$

and

$$cD_j = [cD_j(1)...cDj(N)]$$

N varies between levels.

Wavelet Energy Distribution Ratio
The energy distribution ratio (EDR) for the approximation coefficients at level i is given by

$$EDR_A = \frac{cA_i cA_i^T}{E_T} \tag{2.20}$$

and EDR for detail coefficient j at level i is given by

$$EDR_{D_j} = \frac{cD_j cD_j^T}{E_T} \qquad j = 1, ..., i \tag{2.21}$$

Wavelet Magnitude Coefficients
The magnitude coefficient for the approximation coefficient at level i is calculated by

$$||cA_i|| = \sqrt{cA_i cA_i^T} \tag{2.22}$$

and the magnitude coefficient for the detail coefficient j at level i is calculated by

$$||cD_j|| = \sqrt{cD_j cD_j^T} \qquad j = 1, ..., i \tag{2.23}$$

Wavelet Variance
The variance of the approximation coefficient at level i is calculated by

$$\sigma^2_{cA_i} = \frac{1}{N-1} \sum_{n=1}^{N} (cA_i(n) - \mu_{cA_i})^2 \tag{2.24}$$

and for the detail coefficients by

$$\sigma^2_{cD_j} = \frac{1}{N-1} \sum_{n=1}^{N} (cD_j(n) - \mu_{cD_j})^2 \qquad j = 1, ..., i \tag{2.25}$$

where

$$\mu_{cA_i} = \frac{1}{N} \sum_{n=1}^{N} cA_i(n) \tag{2.26}$$

and

$$\mu_{cD_j} = \frac{1}{N} \sum_{n=1}^{N} cD_j(n) \tag{2.27}$$

2.3 Classification

In this work the classifiers K-nearest neighbors, decision tree, random forest and support vector machine have been implemented. The selection of classifiers was based on the literature presented in section 1.2.1 and considering the relatively small amount of data collected for this work.

2.3.1 K-Nearest Neighbor

K-nearest neighbor (KNN) is a commonly used classifier, it is widely used because of ease of implementation and that it is easy to understand. The classifier saves all the supplied training data and uses it when new entries are classified. The letter K in KNN refers to the number of neighbors used in the classification. When K=1 the training point nearest the testing point will be the only deciding factor when determining the predicted class. If K=3 the three closest points will be deciding the classification, if no distance weight is used the majority vote of the closest points decides the classification. If there are draws the resulting classification can be decided randomly [26, p. 124–126]. In this thesis work a feature set is used as a multidimensional input for each point, neighbors are found by calculating the Euclidean distance. The number of neighbours (K) was set to 3.

2.3.2 Decision Tree

A decision tree is a classifier that uses a tree-like structure. A decision tree starts with one node (the root) which splits the data set into two new nodes based on a split criterion. A split criterion splits the data set depending on whether a feature value is above or below a split value (θ). This process is repeated for every node until a leaf is reached. A leaf is an output for the classification [26, p. 663–666]. An example of a decision tree is shown in Figure 2.10.

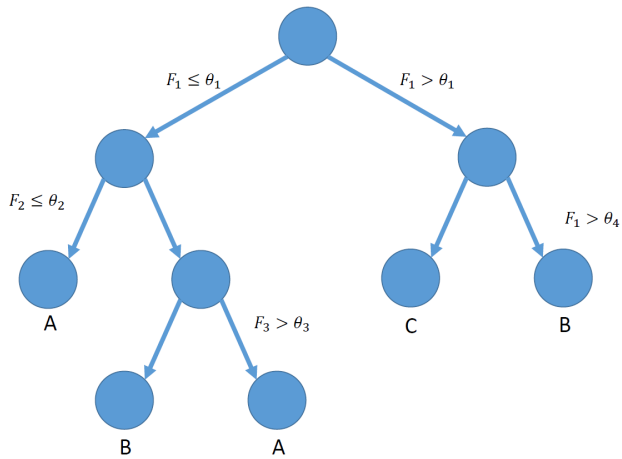

Figure 2.10: A small decision tree with three classes A,B and C as output. The nodes evaluate whether a feature is below or above a threshold value θ_i. The same feature can be used many times with different thresholds, as an example the right branch from the root evaluates feature one (F_1) two times but with different θ.

There are many ways to construct a decision tree, one way is to use the Gini diversity index to evaluate nodes. The Gini index is a measurement of impurity and is calculated as follows

$$i_t = 1 - \sum_j p_j^2 \tag{2.28}$$

where j is the different classes at node t, and p_j is the fraction of class j observed at node t. If only one class is present at a node, the node is pure and the Gini index is zero. When creating node splits they can be evaluated by calculating the weighted impurity gain (ΔI)

$$\Delta I = \frac{n_t}{n} i_t - \frac{n_L}{n} i_{t_L} - \frac{n_R}{n} i_{t_R} \tag{2.29}$$

where n is the total number of samples, n_t samples in node t, i_t is the Gini index for node t, n_L, n_R number of samples in the left and right child node, i_{t_L} and i_{t_R} is the Gini index for the left and right child node. The feature and split value of the feature at each node is decided by calculating the impurity gain for every feature and testing all the values for each feature. By maximizing the impurity gain a new node split can be found. The process of splitting nodes continues until either the maximum allowed number of splits is reached, a proposed split causes the number of observations in a branch node or leaf to be below a pre-determined threshold, no improvement in impurity gain can be found or the node is pure. This approach is a greedy algorithm for construction of the tree [27]. In this work the parameter for maximum number of splits was set to the number of observations minus one. The minimum parent size was set to 10 and the minimum leaf size was set to 1.

Decision trees do often overfit and have high variance. To reduce the effect of overfitting and lower the variance, random forest can be used.

2.3.3 Random Forest

Random forest is the result of a combination of many decision trees. The combination of decision trees often makes the predictions more accurate. The construction of the decision trees starts by creating k new training sets D_i from the original training set D. Each new training set D_i is created by sampling the observations from D randomly with replacement. This means that one observation from D can be selected multiple times in each D_i. From each training set D_i one decision tree is created. The same splitting algorithms used for single decision trees are used in this case as well but instead of using all M features in D_i at each split a random subset of features is used. The subset usually contains $\sqrt{M}$ features. This method is called the random forest algorithm. The class is finally decided by voting. The class with majority outputs from the different trees is the final classification [28][29]. In this work the same parameters used for the decision tree were used for the random forest i.e. the parameter for maximum number of splits was set to the number of observations minus one. The minimum parent size was set to 10 and the minimum leaf size was set to 1. The number of decision trees was set to 100.

Importance of features can be estimated by the help of random forest, this is referred to as random forest variable importance (RF VI). The importance calculation consists of a few steps. Firstly the remaining samples in D, called out-of-bag samples, are classified with the tree constructed from subset D_i and the classification results are saved. Secondly one of the features used in the tree (constructed from D_i) is chosen. The values are then permuted for all out-of-bag samples of that feature. The set including the permuted feature is then classified with the same tree again and the difference in classification result is saved. This process is repeated for all trees. The difference in classification results for the feature are then averaged over all trees and divided by the standard deviation of the score to get a variable importance estimate. This process is repeated for every feature. The acquired variable importance can be useful to find features that are good for separating the data, but it does not give good information about which combinations of features that are useful. Features that are heavily correlated can appear in separate trees and result in high importance scores but when used in combination there is no new information gained [30].

2.3.4 Support Vector Machine

The classifier support vector machine (SVM) was initially constructed for binary classification problems. The original idea of SVM is to separate two classes by finding the hyperplane that separates the classes in the most optimal way. The separation is done using the maximized margin criterion. The goal is to minimize the following equation

$$\min \ \frac{1}{2}\mathbf{w}^{\mathbf{T}}\mathbf{w} + C\sum_{i=1}^{m}\xi_i \ \text{ with respect to } \mathbf{w},\, b \text{ and } \xi_i \tag{2.30}$$

$$\text{subject to } \ y_i(\mathbf{w}^{\mathbf{T}}\phi(\mathbf{F}_i) + b) \geq 1 - \xi_i, \tag{2.31}$$

$$\xi_i \geq 0, \quad i = 1, ..., m$$

where $\mathbf{F_i}$ $(i = 1, ..., m)$ are the feature vectors in the training set of the two classes, y_i is the label of the feature vector ($y_i = -1$ or $y_i = 1$), $\mathbf{w}$ is the weight vector, C is the regularization constant, ξ_i is a slack variable, ϕ is a function that makes a nonlinear mapping of the training data by projecting it into a suitable feature space, see Figure 2.11, and b is a bias term. ξ_i is used for inseparable classes to penalize observations that crosses the margin boundary for their class. If an observation do not cross the boundary ξ_i equals 0. C is a parameter used to prevent overfitting and controls the maximum penalty added to observations violating the margin [31]. A small value of C results in a large margin and a large value of C results in a narrow margin. The optimal margin for separable classes equals $\frac{2}{||\mathbf{w}||}$. To get a better understanding of SVM see Figure 2.12.

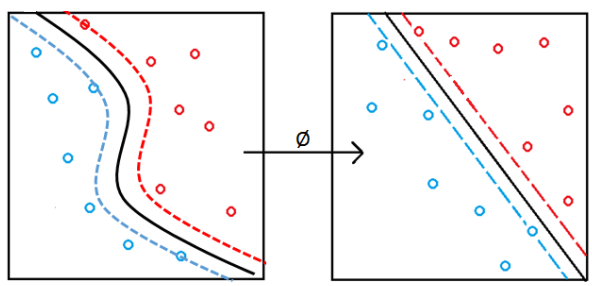

Figure 2.11: Illustration of the function ϕ.

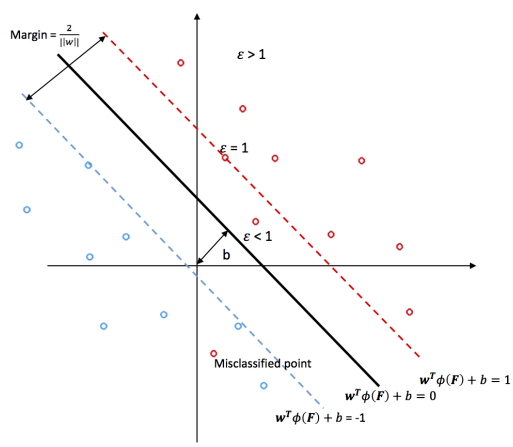

Figure 2.12: Description of SVM.

Lagrange multipliers method is a common way to solve the optimization problem. The following equation is minimized

$$\min \ \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \mathbf{F}_i'\mathbf{F}_j - \sum_{i=1}^{m}\alpha_i \quad \text{with respect to } \alpha_i\text{'s} \tag{2.32}$$

$$\text{subject to} \quad \sum_{i=1}^{m}\alpha_i y_i = 0 \tag{2.33}$$

$$0 \leq \alpha_i \leq C, \quad i = 1, ..., m$$

where α_i's are the Lagrange multipliers [32].

Commonly used methods to solve multi-class problems with SVM are the one-versus-one technique and the one-versus-rest technique. Both methods use the standard binary SVM multiple times in a systematic way to solve the classification problem.

2.3.4.1 One-versus-One

The one-versus-one approach evaluates all possible combinations of binary classifiers which results in $\frac{k(k-1)}{2}$ classifiers for a k class problem. The winning class for every combination for a test observation gets one vote and the class with most votes is the final label of the test observation [31].

2.3.4.2 One-versus-Rest

The one-versus-rest technique constructs k binary classifiers to label a test observation in a k class problem. The l-th classifier is trained by using the training data for the l-th class as the positive class ($y = 1$) and the other $k-1$ classes as the negative class ($y = -1$). The class of a test observation is decided by the classifier with the highest output value. The binary SVM classifier is sensitive to highly unbalanced data and the one-versus-rest approach can thereby be problematic [31].

2.4 Feature Selection

The goal of feature selection is to find a limited number of features, from the set of extracted features, which yield the smallest classification error possible. The number of features included can be chosen beforehand or by some criterion.

2.4.1 Sequential Forward Feature Selection

Sequential forward feature selection (SFFS) is a simple and relatively fast method for feature selection. SFFS starts with an empty feature set. Features are then added to the feature set one by one by using some evaluation method of the classification error. In this work, the classification error is calculated as the number of misclassified windows relative to total number of windows. During the first iteration, the single feature that yields the smallest classification error is added to the empty feature set. In the second iteration a new feature from the remaining feature set is added

to the feature set. The chosen feature is the feature that together with the first feature yields the smallest classification error. New features are then added to the feature set using the same method. The feature set is complete when there is no improvement of the classification error when a new feature is added to the set, when the improvement is below a selected threshold or when a pre-selected number of features have been added to the feature set [33]. In this work the classifiers KNN and SVM were used. The classification error was the average misclassification rate when leave-one-out cross validation was performed. In leave-one-out cross validation one observation (riding session) is used as validation set and the other sets are used as training set and the classification error is calculated. This is repeated until all sets have been used as validation set. The average error can then be calculated.

2.5 Feature Reduction

2.5.1 Principal Component Analysis

Principal component analysis (PCA) is a method used for dimensionality reduction. PCA is an orthogonal projection of data onto a lower dimensional linear space such that the variance of the projected points is maximized and the data becomes uncorrelated. The principal component is the component with the largest variance and the last component has the smallest variance. In activity recognition PCA can be used to reduce the number of features used for classification of the data. The first step in PCA is that the mean of each feature vector in the feature set is calculated and subtracted from the feature values. The feature vectors are columns in the feature matrix $\mathbf{F}$ (containing N features).

$$\mathbf{F}_{\mu_\mathbf{F}} = \mathbf{F} - \mu_\mathbf{F} \tag{2.34}$$

The covariance matrix of the new vectors is calculated. Eigenvalues and eigenvectors of the covariance matrix are calculated in the following way

$$cov(\mathbf{F}_{\mu_\mathbf{F}})\mathbf{V} = \lambda\mathbf{V} \tag{2.35}$$

where $cov(\mathbf{F}_{\mu_\mathbf{F}})$ is the covariance matrix of the feature set, $\mathbf{V}$ are the eigenvectors $(v_1, v_2, ..., v_N)$ and λ the eigenvalues in a diagonal matrix. The eigenvalues are sorted in descending order and a matrix is received by the corresponding eigenvectors. The eigenvector corresponding to the largest eigenvalue is the principal component (the first column) and the eigenvector corresponding to the smallest eigenvalue is the last component (the last column). A transformation matrix is obtained by keeping the p first eigenvectors. Usually the p chosen eigenvectors represent 95 % of the variance of the original feature set. The dimensionality of the original feature set can now be reduced by the use of the transformation matrix $\mathbf{V_p}$.

$$\text{Reduced feature set } = \mathbf{F}_{\mu_\mathbf{F}}\mathbf{V_p}, \quad \text{where } \mathbf{V_p} = v_1, v_2, ..., v_p \tag{2.36}$$

The new feature set has now p dimensions. Observe that the new feature set obtained from PCA is not the initial feature set but a linear combination of the initial features [34][35].

3. Data Collection

This chapter describes different choices made for the data collection and how the data was collected.

3.1 Sensors

3.1.1 Description of Sensors

In this work equine activity recognition data were collected using accelerometer and gyroscope. Examples of the different sensor signals for the x, y, z-axis and the magnitude are seen in Figure 3.1. The choice of sensor types was based on the survey of the field, the available resources provided by Sony and the desire to be able to use cheap consumer products for data collection and analysis. The accelerometer and gyroscope used are therefore units in Sony smartphones. For the collections of data two different phones were used, both containing the accelerometer unit BMA2X2 and the gyroscope unit BMG160. The sensor range for the accelerometer was ± 39.2 m/s^2 and for the gyroscope ± 34.9 rad/s. An internal prototype device containing an accelerometer component was also considered but was discarded due to not having easy access to the data collected by the unit.

To access the data from the accelerometer and gyroscope different options were examined. Different applications available at Google Play were briefly tested if they could suit the needs for the data collection. For example Sensor Kinetics [36] was tested but limitations due to not allowing data extraction in the free version it was discarded. The possibility to use in-house developed applications for data was discussed but no such application with easy access to data was found. The final choice of application to use resulted in AndroSensor available at Google Play [37]. This application had some interesting features such as remote start and stop, and could be configured to write the collected data into a comma-separated value (CSV) file format.

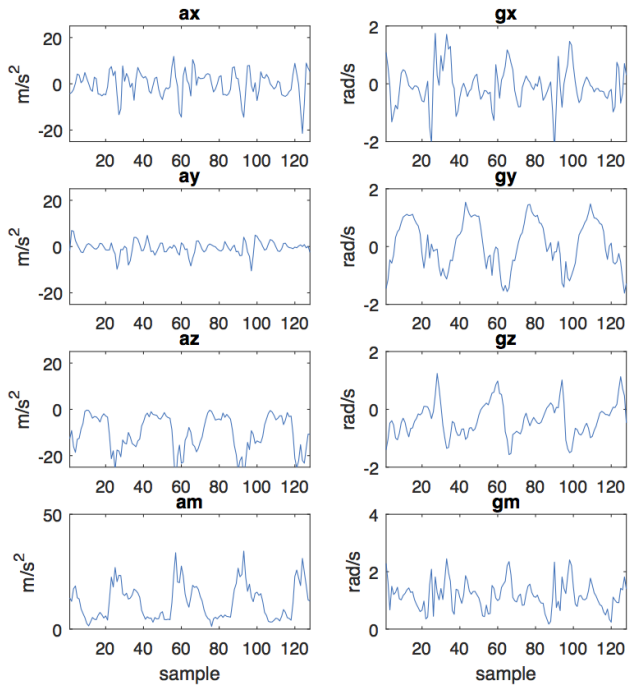

Figure 3.1: An example of collected accelerometer and gyroscope signals for all axes, including the calculated magnitude, from one riding session. The sample frequency is 50 Hz, the gait is canter and the data is collected by a phone placed around the saddle-girth.

3.1.2 Sensor Placement

The sensor placement when analyzing motion patterns plays an important role. A final decision was made to place a sensor at the saddle-girth and one sensor on the back of the rider. The sensor placements were based on the literature discussed in section 1.2.1.2 and section 1.2.2, consumer products and interviews. It was important that the smartphones were easy to access and attach, would stay at the same position during the whole session, and would not affect the riding pattern of the horse.

Initially there were many ideas for sensor positions. The positions were, on the head of the horse, at the back of the horse, in the saddle, on the legs, on the rider, at the saddle-girth and at the sternum of the horse. The head of the horse was quickly discarded as a location due to problems with attaching the phone and the weight of the phone would probably be an annoyance for the horse. Similarly, the sternum and back of the horse were discarded as locations due to needing extra equipment that was not always used during riding. The option to place the sensors at the legs of the horses would probably require sensors with a wider accelerometer range than available in the phones, and also the weight of the phones could be a disturbance.

The best placement on the horse thus resulted in using the saddle-girth. The girth keeps the saddle in place and everyone that uses a saddle has one. This location had also been used in some of the research and products studied (see section 1.2.2). In an interview with Lisette Hedman, who titles herself as 'horse and riding mechanic', these locations were further verified as a good location for gait analysis. To fixate the sensor (smartphone) a sports activity armband was fastened around the saddle-girth, on the part of the girth that is under the belly of the horse. The screen of the phone was facing downwards and the top of the phone was pointing to the right side of the horse.

A pilot data collection with the suggested sensor location at the girth was done, an example of annotated data from this study can be seen in Figure 3.2.

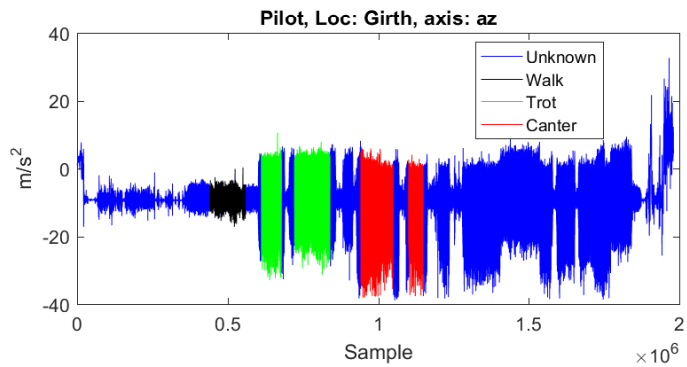

Figure 3.2: The full pilot data signal collected. The signal is the z-axis of the accelerometer. The sensor was placed on the saddle-girth. Regions where annotations were made are colored for the respective activity.

A sensor was decided to be placed on the rider as well. This was done because the rider moves differently depending on the gait and some other consumer products, for example Equilab, uses a pocket on the rider as position for the sensor in activity recognition analysis. To make the data collection reproducible for different riders and for minimal sensor movement a sport waist pouch was used to place the phone on the lower back. The back of the phone was against the back of the rider and the top of the phone was pointing to the right side of the back.

3.2 Collection of Training Data

The pilot data collection was successful and a training data set was collected with the chosen sensor positions, see Table 3.1. The data set was collected in collaboration with staff and students at Flyinge Kungsgård and with an employee at Sony. The horses used for collection did vary in size and gender.

Table 3.1: Pilot and training data collected

Rider	Horse	Location	Duration	Date	Notes
RA	HA	Large Riding House, Flyinge	30 min	20/2	indoor, pilot data
RB	HB	Racecourse, Flyinge	1 h	10/3	outdoor
RC	HC	Crafoord, Flyinge	40 min	14/3	indoor
RD	HC	Crafoord, Flyinge	50 min	14/3	indoor
RC	HD	Crafoord, Flyinge	55 min	14/3	indoor
RE	HE	Riding Stable, Genarp	55 min	9/3	indoor

A protocol for recording the data was constructed and followed. The protocol defined positions for the two different phones, settings in AndroSensor, procedure at start of the recording, how annotation of data was made and what other parameters to be noted e.g. location, horse and rider.

The phones were always placed in the same position with the same orientation. The sampling rate was set to 50 Hz in AndroSensor. Before fastening the two phones at their corresponding positions, recordings were started on both devices. Additionally, a stopwatch was started on a third device. All three devices were started at the same time manually. The third device with a stopwatch was used to get timestamps when new activities were started, a new lap meant that a new activity was started. Manual annotation of all activities and the lap number were made. When the session ended the stopwatch was stopped and all the timestamps were saved. The phones used for recording data were stopped manually. In some recordings, a forth device, a camera, was used to videotape the session. The video was used in the labeling when something was unclear. The time of the camera was synchronized by videotaping the stopwatch.

As mentioned earlier AndroSensor had the option for remote start by sending a SMS to the phone. This option was not used in the end due to problems with messages not arriving at the same time for the phones and no confirmation of recording start was received.

The final training data set included 4 different horses and 4 different riders from five different sessions. The pilot data set only contained one of the sensor positions and was, because of this, excluded from the training set.

4. Preprocessing

This chapter describes the process of preparing the collected training data for the analysis made in the following chapters.

4.1 Matlab Implementation

A work flow in Matlab was developed for analyzing the collected data. The work flow was divided into several smaller segments, with three major sections preprocessing, feature extraction and classification, see Figure 4.1. Important parts of these sections are described in this chapter and the two following chapters.

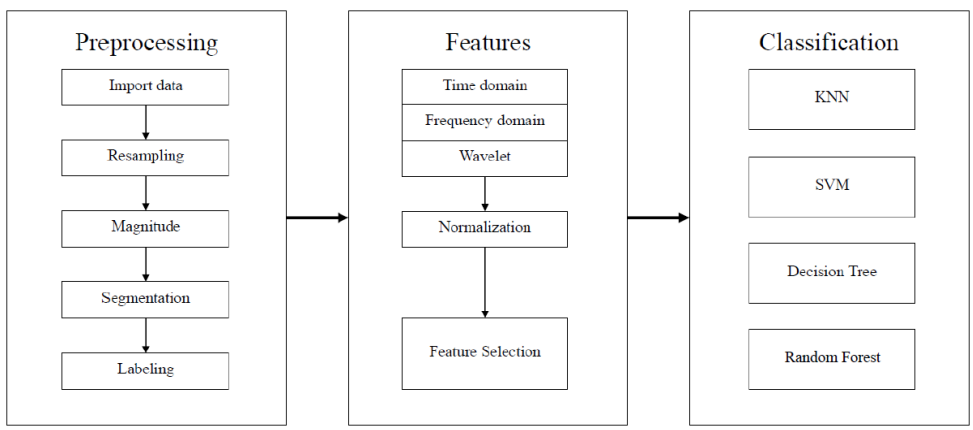

Figure 4.1: Simplified overview of the implemented Matlab work flow.

4.1.1 Preprocessing

4.1.1.1 Importing Data into Matlab

The output from the application AndroSensor generated CSV-files with columns for the x, y, z-axes for the accelerometer and the gyroscope, respectively.

The data from both sensor positions for all subjects were imported into Matlab with a set of implemented functions. The functions loaded the data from the recorded CSV-files (specific to the collected CSV-files) into a flexible Matlab format. The code and format supported the possibility to import data from several 'time-synchronized' sensor positions from a single subject. Each sensor position could contain several sensors (accelerometer, gyroscope et cetera) under the assumption that the sensors had the same number of samples and the same timestamps.

4.1.1.2 Resampling

There were some fluctuations in the sampling of the data and the collected data was not perfectly evenly sampled. All collected data was therefore resampled to have

the same time distance between samples. With the sampling rate set to 50 Hz, the time difference between samples were 20 ms. The resampling was done because the time difference would otherwise create problems in the feature extraction and in the segmentation of the data.

Two different methods for resampling were tested, both methods used Matlab's built-in function *resample*. The different methods were linear interpolation and cubic spline interpolation. The result of both methods were visually examined, both methods seemed to perform well, but the cubic spline would sometimes create a very large deviation from the signal, an example of this can be seen in Figure 4.2. It was thereby decided to use the linear interpolation method as it looked good when it was visually examined and seemed to have less problems than cubic spline interpolation method.

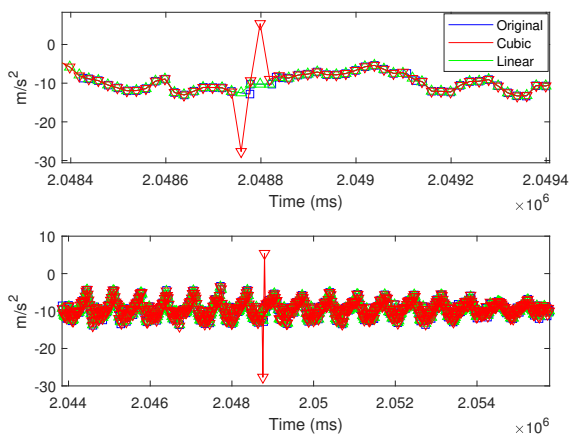

Figure 4.2: Example of the interpolation methods. The plots illustrates a large deviation from the signal for the cubic spline interpolation method. The top plot is a zoom in of the signal where the deviation appears. The bottom plot is a zoom out of the signal.

4.1.1.3 Magnitude

A commonly used method in activity recognition is to construct a magnitude vector from the collected data. The magnitude vector was calculated for both the accelerometer and gyroscope signals as follows

$$a_m(n) = \sqrt{a_x(n)^2 + a_y(n)^2 + a_z(n)^2} \tag{4.1}$$

$$g_m(n) = \sqrt{g_x(n)^2 + g_y(n)^2 + g_z(n)^2} \tag{4.2}$$

The vectors $a_m(n)$ and $g_m(n)$ was used as additional dimensions for the features. The total number of dimensions used for feature extraction were thereby 8 ($a_x(n)$, $a_y(n)$, $a_z(n)$, $a_m(n)$, $g_x(n)$, $g_y(n)$, $g_z(n)$ and $g_m(n)$).

4.1.1.4 Segmentation

A function for automatic segmentation of the data collected was implemented. The method is able to create windows of the collected data with specific number of samples and overlap between windows. For each window the timestamps of the samples were also kept. When approaching the end of the signal the number of samples left might be too few for a full window, if that was the case the window was discarded.

Since the sensors were manually started and stopped, the synchronization between them was not perfect. One sensor signal would sometimes fit more windows than the other sensor signal. If any sensor signal had more windows than the other, the sensor with the fewest windows decided the maximum number of windows extracted. The segmentation was used in conjunction with labeling to extract the interesting parts of the signal.

4.1.1.5 Labeling

During the data collection manual annotation of activities and time stamps were recorded. For every activity recorded a start time and an end time were obtained. Usually the end time would be the same as the next activity start time, but due to human errors this was not always the case. The annotated data was paired with the collected sensor data to label the extracted windows. Every window that fit between the start time and end time of an activity was labeled as the recorded activity. Windows not fitting between any start times and end times was labeled as 'Unknown'.

For all sensor placements, the assumption was made that the recording sensors are synchronized, or close enough to have the same timestamps for the extracted windows. This means that window n from a sensor position gets the same label as window n from another sensor position.

4.1.1.6 Outlier Rejection

Data with unknown label and data sections that deviated a lot from the signal in general were rejected. Moreover, two different methods to handle possible outlier values were briefly tested but in the end not used. The methods constructed thresholds based on the measurements standard deviation and median absolute deviation and values above the thresholds were flagged. Both methods flagged for outliers in many of the windows where no clear outliers could be seen by the eye. It was concluded that no outlier rejection was needed.

4.1.2 Decision of Sensor Position

Before the training data set was created the decision was made to only use the sensor position on the saddle-girth of the horse. This decision was made after visual examination of the different signals. The accelerometer signal for the phone placed on the riders back did saturate, in contrary to the phone placed on the saddle-girth.

4.1.3 Training Data Set

By performing all the steps above and removing all windows with unknown label, two data sets for the girth sensor were constructed from the data. The difference between the data sets results from the selected window length when applying the segmentation function and the labeling. One of the data sets uses a window length of 128 samples (2.56 seconds) and the other set uses a window length of 256 samples (5.12 seconds). Both the sets used a 50 % overlap between windows. The decisions of window lengths and overlap were based on the articles described in section 1.2.1. The total number of windows for each activity for both sets is presented in Table 4.1, for an example see Figure 4.3.

Table 4.1: The table shows the number of windows for each class in the constructed training set for window lengths with 128 and 256 samples using 50 % overlap.

Window length	128		256	
Stand	251	(2.99%)	102	(2.51%)
Walk	3825	(45.61%)	1857	(45.65%)
Trot	2653	(31.63%)	1299	(31.93%)
Canter	1658	(19.77%)	810	(19.91%)
Total	8387	(100.00%)	4068	(100.00%)

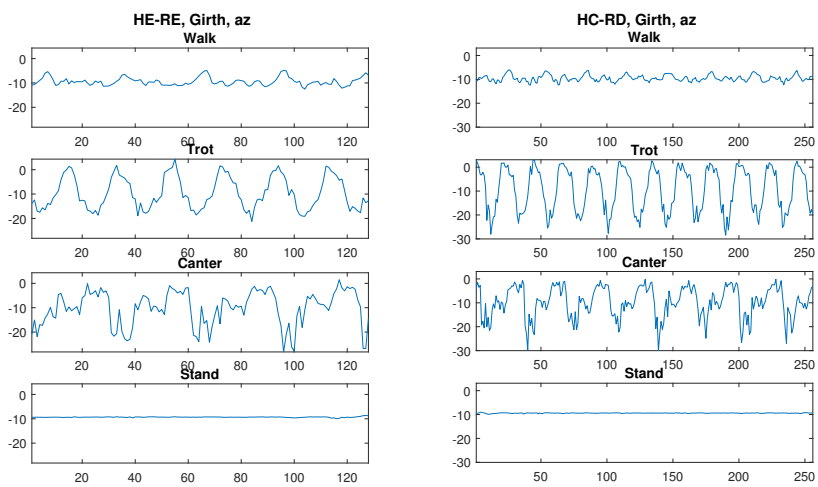

Figure 4.3: Example of windows extracted from the data set, one window of each labeled activity from HE-RE (left) and HC-RD (right). The windows are constructed with a window length of 128 (left) and 256 (right) samples. The data is from the z-axis of the accelerometer signal.

5. Algorithm Development

This section describes the development of the equine gait recognition algorithm. Several different algorithms were developed and evaluated using different feature sets and classifiers before a final algorithm was decided. An overview of the algorithm development process is visualized in Figure 5.1.

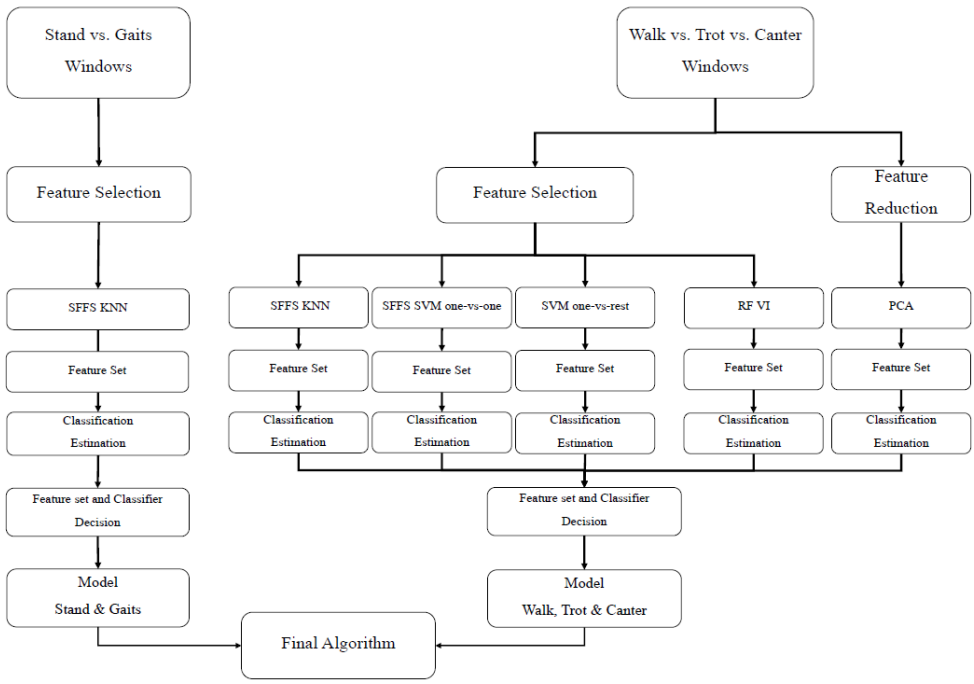

Figure 5.1: An overview of the development of the equine gait recognition algorithm. The development is described in this chapter.

5.1 Feature Extraction

Feature extraction was done using Matlab. All implemented features have earlier been used in human and equine activity recognition, see section 1.2.1.

The time domain, frequency domain and wavelet based features listed in Table 5.1 were implemented and calculated for all windows. The total number of features implemented was 405. All features were calculated for the window lengths 128 and 256 samples, respectively.

Table 5.1: List of Features

(a) Time Domain Features

Mean (Eq. 2.1)
Median
Quartile 1 (Fig. 2.3)
Quartile 2 (Fig. 2.3)
Quartile 3 (Fig. 2.3)
Interquartile range (Fig. 2.3)
Standard Deviation (Eq. 2.3)
Skewness (Eq. 2.4)
Kurtosis (Eq. 2.5)
Root Mean Square (Eq. 2.6)
Crest Factor (Eq. 2.7)
Energy (Eq. 2.8)
Range (Max-Min) (Fig. 2.6)
Mean-Crossings (Fig. 2.7)
Maximum of Differences (Eq. 2.9)
Max Cross-Correlation (Eq. 2.10)
Correlation Coefficient (Eq. 2.11)

(b) Frequency Domain Features

Sum Largest Fourier Coefficients (Eq. 2.14)
Dominant Frequency (Fig. 2.8)
Entropy (Eq. 2.15)
Quartile 1
Quartile 2
Quartile 3
Interquartile range

(c) Wavelet Based Features

EDR A Coefficients (Eq. 2.20)
EDR D Coefficients (Eq. 2.21)
Magnitude A Coefficients (Eq. 2.22)
Magnitude D Coefficients (Eq. 2.23)
Variance A Coefficients (Eq. 2.24)
Variance D Coefficients (Eq. 2.25)

All features in Table 5.1, except the features maximum of differences, max cross-correlation and correlation coefficients were calculated for all eight different axes (a_x, a_y, a_z, a_m, g_x, g_y, g_z and g_m). The features maximum of differences, max cross-correlation and correlation coefficients were instead calculated for pairs of axes. The different combinations of pairs were a_z and a_x, a_z and a_y, a_x and a_y, g_z and g_x, g_z and g_y, g_x and g_y and a_m and g_m.

For the wavelet based features Daubechies 4 was chosen as mother wavelet since it is a commonly used wavelet for motion data (see e.g. [12]). The level was chosen to 5 after evaluating the different levels based on classification accuracy of the training data using only the wavelet feature set. For details on calculation of the implemented time, frequency and wavelet features, see section 2.2.

5.1.1 Normalization

All calculated features were then normalized by the following equation

$$F_{N_i} = \frac{F_i - \mu_i}{\sigma_i} \tag{5.1}$$

where F_i is the feature number i, μ_i is the mean for the i-th feature in the data set and σ_i is the standard deviation for the feature i in the data set.

5.2 Decision of Window Length

The decision was made to use the window length 128 samples (2.56 seconds). This decision was made after studying feature values and after evaluating some classification results, using the training data, for the different window lengths (128 and

256 samples). The decision to use the shorter window length was also based on the riding lessons studied. It is not unusual that the rider does rapid changes between the gaits and a longer window would not capture some of these changes. In the proceeding development of the algorithm only data from the saddle-girth and windows with 128 samples is used.

5.3 Classify Stand vs. the Gaits

During the algorithm development process it was noticed that the data labeled "Stand" often got misclassified even though these signals differs a lot from the different gait signals (see Figure 4.3). The reason for this problem was probably because the data set was very unbalanced, less than 3 % of the total data set was labeled as "Stand". The feature selection methods did thereby not focus on finding features for correct classification of "Stand". To avoid this problem features that can be used to distinguish between stand and all the gaits as one joint class were selected as a first step.

5.3.1 Feature Selection

Sequential forward feature selection (SFFS) using KNN as classifier was used as selection method. As described above, whenever KNN are used in this project K=3. This method was chosen since it is simple and relatively fast and good results were obtained when it was tested. The selection was made using leave-one-out cross validation. One riding session of the training data was used as test set and the other four riding sessions of the training data were used as training set. This was repeated five times, i.e. all sessions were left out once, and the average classification result was calculated. The classification error improvement threshold was set to 0.01 %, i.e. if the improvement was lower then 0.01 % the feature was not added to the set and the set was complete. The low threshold was selected since the data set was very unbalanced. The selected features are listed in Table 5.2 and visualized in Figure 5.2. In Figure 5.2 it can be seen that the feature on the x-axis (RMS) could be good alone. It seems like the feature on the y-axis (energy) has a very small contribution to the separation. Regardless, both features were used since they were chosen by the selection method.

In the following tables and figures, the first letter stands for time, mixed axes time, frequency or wavelet feature (T, M, F or W), the following letters stands for accelerometer or gyroscope (a or g) and whether it is x, y, z-axis or the magnitude (x, y, z or m).

Table 5.2: The selected features to distinguish stand from the gaits for window length 128 and sensor position girth.

T gm: Root Mean Square
T ax: Energy

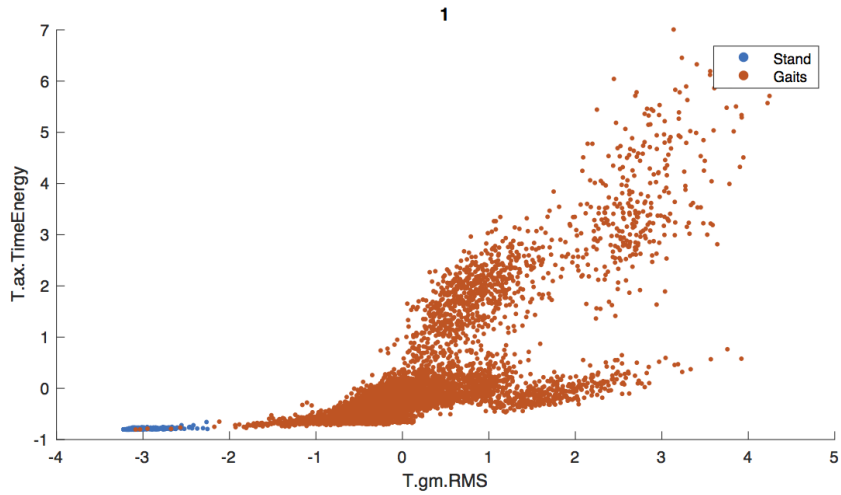

Figure 5.2: Scatter plot of the two selected features to distinguish stand from the gaits. In the scatter plot feature one (x-axis) is plotted against feature two (y-axis).

To distinguish between stand and the gaits, as a joint class, only the KNN classifier was used and trained using the feature set listed in Table 5.2. The classification result was estimated by using leave-one-out cross validation. The estimated result of the feature set using the training data was 99.9 %.

5.4 Classify Walk vs. Trot vs. Canter

The next step was to find features to distinguish between the different gaits, i.e. walk, trot and canter. Different feature selection and feature reduction methods were tested to find the best feature set for the final algorithm. These methods were sequential forward feature selection using KNN, SVM one-vs-one and SVM one-vs-rest, feature selection based on RF VI and feature reduction using PCA.

Five different classifiers were used to evaluated the selected feature sets, by the above mentioned methods, and to find the best classifier for the final algorithm. The classifiers were KNN, decision tree, random forest, SVM one-vs-one and SVM one-vs-rest. Leave-one-out cross validation was used to get an estimation of the classification performance of the feature sets.

In the previous section only one selection method (SFFS KNN) and one classifier (KNN) were used to distinguish between stand and all the gaits as one joint class. In this section five different selection methods and five classifiers have been evaluated to distinguish between the gaits walk, trot and canter. The reason is that the gait classification problem is more complex and the main focus in this project has been to classify different gaits.

5.4.1 Feature Selection

5.4.1.1 Sequential Forward Feature Selection using KNN

Sequential forward feature selection using KNN was used as selection method in this step as well. The selection was made using leave-one-out cross validation where the different riding sessions were left out one by one. The classification error improvement threshold was set to 0.5 %. The selected features are listed in Table 5.3 (a) and the estimated classification result of that feature set using leave-one-out cross validation of the training data is presented in Table 5.3 (b). The classification result of the random forest changes slightly from time to time since random features are chosen when creating the decision trees and thereby the result is rounded to include less significant figures.

Table 5.3: SFFS KNN

(a) Selected Features

W az: EDR cD_4
F gz: Dominant Frequency
T az: Skewness

(b) Estimation of Classification

KNN	99.1 %
Decision Tree	98.8 %
Random Forest	∼99 %
SVM one-vs-one	97.1 %
SVM one-vs-rest	96.7 %

5.4.1.2 Sequential Forward Feature Selection using SVM One-vs-One

Sequential forward feature selection using SVM one-vs-one as classifier was also done using 0.5 % as classification error improvement threshold and leave-one-out cross validation. The selected features are listed in Table 5.4 (a) and the estimation of the classification result is listed in Table 5.4 (b).

Table 5.4: SFFS SVM one-vs-one

(a) Selected Features

F gz: Dominant Frequency
T az: Interquartile range

(b) Estimation of Classification

KNN	97.4 %
Decision Tree	98.5 %
Random Forest	∼97 %
SVM one-vs-one	99.2 %
SVM one-vs-rest	96.6 %

5.4.1.3 Sequential Forward Feature Selection using SVM One-vs-Rest

Sequential forward feature selection using SVM one-vs-rest was done in the same way as the above sequential forward feature selection methods and the results are seen in Table 5.5.

Table 5.5: SFFS SVM one-vs-rest

(a) Selected Features

W az: EDR cD_4
T az: Skewness
F az: Largest Fourier Coefficient
F gy: Sum 2 Largest Fourier Coefficients

(b) Estimation of Classification

KNN	99.4 %
Decision Tree	98.4 %
Random Forest	∼99 %
SVM one-vs-one	99.6 %
SVM one-vs-rest	99.4 %

5.4.1.4 Random Forest Variable Importance

The 20 features with the highest importance in the random forest were selected. The whole training set was used as input when the trees were created and the importance was decided. The importance was calculated as described in section 2.3.3. All 20 selected features are listed in Table 5.6 (a). It should be noted that the features in the table may change slightly due to the random construction of the trees. The estimated classification result of the set is seen in Table 5.6 (b).

Table 5.6: Random Forest Variable Importance

(a) Selected Features

F gy: Dominant Frequency
F am: Dominant Frequency
F az: Dominant Frequency
W gy: Magnitude Coefficient cD_5
W ax: Magnitude Coefficient cD_4
T az: Skewness
M gz gy: Max Cross-Correlation
W gy: Variance cD_5
F gz: Dominant Frequency
F gy: Largest Fourier Coefficient
W ax: EDR cD_4
W ax: Variance cD_4
M az ax: Correlation Coefficient
W az: EDR cD_4
M gz gy: Correlation Coefficient
T am: Mean-Crossings
M gz gx: Correlation Coefficient
W az: Variance cD_4
T az: Median
W az: Magnitude Coefficient cD_5

(b) Estimation of Classification

KNN	98.4 %
Decision Tree	90.7 %
Random Forest	∼97 %
SVM one-vs-one	98.6 %
SVM one-vs-rest	99.0 %

5.4.2 Feature Reduction

5.4.2.1 PCA

The principal components which describe 95 % of the variance were used as transformation matrix to reduce the feature set. The feature set was reduced from 405

dimensions to 63 dimensions. The estimated classification result for training data of the new feature set is presented in Table 5.7.

Table 5.7: Estimation of classification result for training data using the reduced feature set.

KNN	94.4 %
Decision Tree	84.9 %
Random Forest	∼95 %
SVM one-vs-one	98.2 %
SVM one-vs-rest	97.8%

5.5 Final Gait Recognition Algorithm

The choice of the final model to use for classification of walk, trot and canter was KNN using the feature set acquired by SFFS using KNN. This model in combination with the model separating stand and the gaits (based on features from SFFS KNN and classifier KNN) resulted in the final algorithm, see Figure 5.3 for an overview.

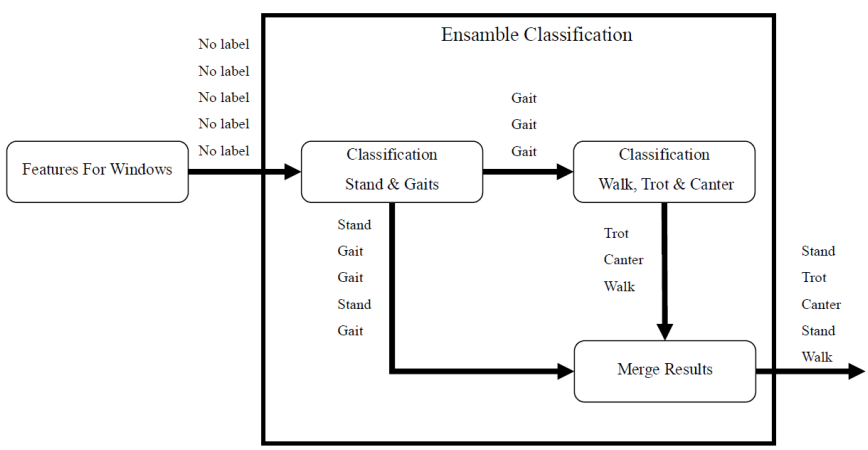

Figure 5.3: Illustration of the classification process for segmented data with no label. The selected features for each segment are sent to the first classifier which classify stand and gaits, the windows classified as gaits are sent to the next classifier which classify the type of gait using the feature set selected for gait classification. The results are then merged together and a classification for each window is achieved.

The selection of feature set and classifier for the final algorithm was done by analyzing the results of the different selection methods and the classifiers, see Table 5.3–5.7. The reason PCA was not used is because it is mainly used to reduce the size of feature sets. However, none of the extracted feature sets were large enough to justify the use of PCA on them. PCA was used on the full feature set and reduced the set

to 63 new features. The amount of computation needed to calculate the full feature set, which is needed when calculating the new feature set, is one drawback of using PCA. To be able to get a good feature set more evaluation of the new features are needed and since good results were obtained with the other feature sets, this was not further evaluated in this work.

The feature set based on RF VI was not used since the importance score does not consider the correlation between features. This means that for example the best two features could be very good on their own and in combination with other features, but when used together no new or very little new information is gained. Further evaluation of feature combinations will be needed.

All the feature sets selected by the SFFS methods were small (2-4 features) and the classification results when using leave-one-out cross validation were high. These feature sets and classifiers would probably all have performed well on unseen data. The reason the feature set from SFFS using KNN was used for the final algorithm was because these features were found in the other SFFS feature sets, one feature is found in the feature set constructed by SFFS using SVM one-vs-one and two features in the set constructed by SFFS using SVM one-vs-rest. All the three selected features also appear in the list of the features with highest importance in the random forest. In Figure 5.4 a visual representation of the selected feature values of the training data is shown. The classifier with the best result for this SFFS KNN set was KNN and thereby it was decided to use KNN as classifier in the final algorithm.

Another thing to note is that all selected feature sets contains features from both the accelerometer and the gyroscope. All different domains (time, frequency and wavelet) are also represented in the feature sets were at least three features are selected.

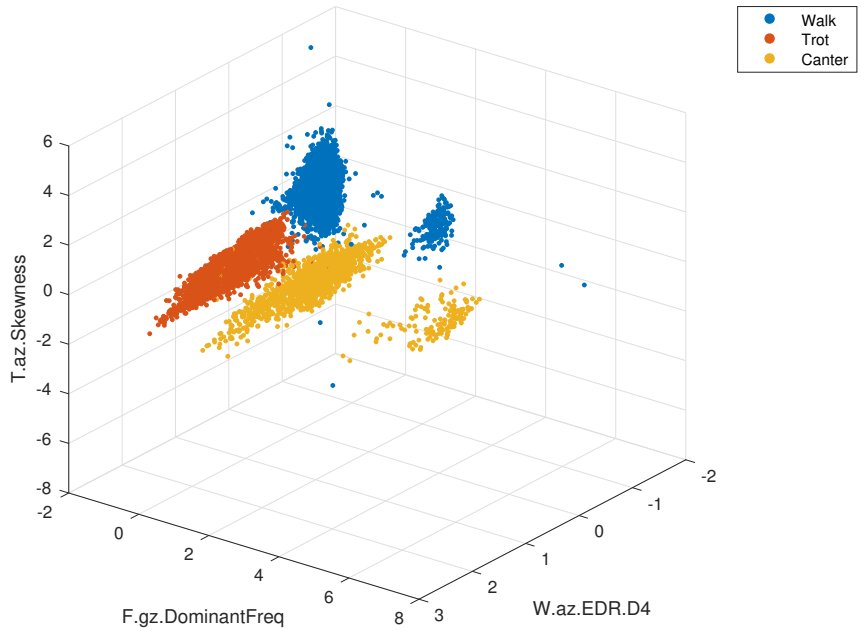

Figure 5.4: Three dimensional scatter plot of the features selected to distinguish between the gaits in the final algorithm.

6. Result

In this chapter the process of collecting and preprocessing data for testing the algorithm is described. The algorithm is evaluated and compared to the smartphone application Equilab.

6.1 Collection of Testing Data

The data used for evaluation was collected in the same way as the training data and the same phone was used. The phone was placed on the saddle-girth. Accelerometer and gyroscope data were collected using the application AndroSensor. The data was collected during two different dressage lessons outdoor at Flyinge Kungsgård. Two different riders were used for evaluation and both riders were riding for approximately 50 minutes. To get a better evaluation of the performance of this work the result is benchmarked against the existing smartphone application Equilab. Another phone, with Equilab downloaded, was thereby placed on the rider. Equilab was started the same time as the riding lesson started and stopped when it ended. The phone had the accelerometer MPU6500 Acceleration Sensor Invensense with maximum range $\pm 39.2\ m/s^2$ and the gyroscope MPU6500 Gyroscope Sensor Invensense with maximum range $\pm 34.9\ rad/s$. One rider wore the phone with Equilab in a tight jacket pocket and the other rider wore the phone in the pocket of her trousers, see Table 6.1 for information about the riding sessions.

Table 6.1: Testing data collected at Flyinge

Session	Rider	Horse	Location	Duration	Date	Notes	Equilab
Test 1	RF	HF	Kastanjegården, Flyinge	48 min	15/5	outdoor	jacket
Test 2	RG	HG	Kastanjegården, Flyinge	49 min	15/5	outdoor	trousers

6.2 Preprocessing of Testing Data

The processing of the testing data was a bit different compared to the training data. The first steps were the same with resampling the signal and calculating magnitude vectors. The next step was to extract the labeled part of the signal from the recorded signal, i.e. the data from the dressage lesson. The extracted signal was then segmented with the implemented segmentation function. The windows created then had their features calculated. The calculated features were then sent to the chosen classifier to get a label for the windows.

The ground truth for the signal was defined for each sample of the analyzed signals. This meant that the predicted labels for the windows had to be converted to labels corresponding the samples covered by the window. The overlap between windows meant that a sample could possibly receive two different labels, the final classification of a sample was chosen to be the latest window classifying the point, see Figure 6.1.

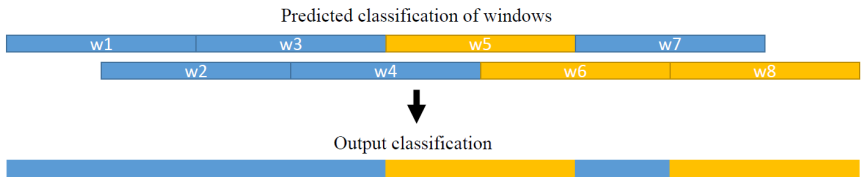

Figure 6.1: An illustration of how the final output classification is constructed. Each window has its own classification and the number of samples covered by the window get the same label. When windows are overlapping with different labels, the latest windows decide the labels for the samples covered by the windows.

6.3 Classification Result

6.3.1 Result Test Session 1

The classification accuracy of test session 1 with the developed algorithm is 94.1 % and can be found in Figure 6.2. In Figure 6.3 a detailed view of where the errors appear can be seen. The middle line marks the errors between the predicted label and the expected label. Many of these errors are from the projection of the window label to sample based labeling (see Figure 6.1) when activity change occurs, each single sample classified wrong creates an error marker. The result of the developed algorithm is benchmarked against the smartphone application Equilab. The classification result of the developed algorithm is translated to times and compared to the true times and the times presented by Equilab in Table 6.2.

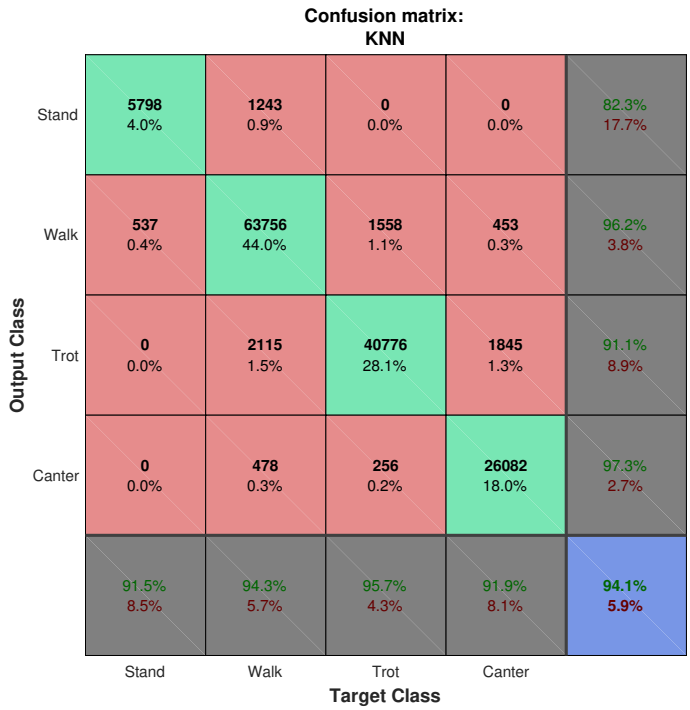

Figure 6.2: Confusion matrix of the result from test session 1. The overall classification accuracy is presented in the bottom right square. The output class is the result from the developed algorithm and the target class is the expected result. The percentage seen in the green and the red squares are the amount of samples classified in that square. The right column in grey describes the precision, i.e. how many of the predicted samples for the label were correctly classified. The bottom row describes the recall, i.e. for all samples that should have been the label, how many of these were found.

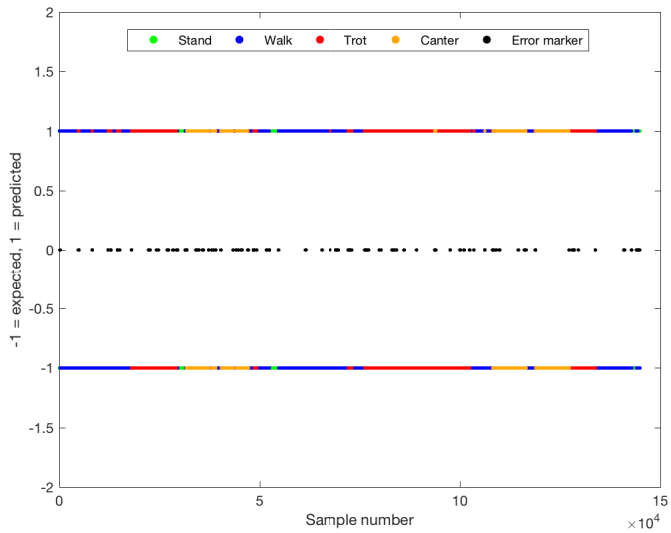

Figure 6.3: The plot visualizes the comparison of the expected (the line at -1) and the predicted (the line at 1) result from test session 1. The black marks (line at 0) visualize the errors. Note that the error marks are approximately 6 % of the expected and predicted lines even though this can not be seen by the eye.

Table 6.2: Comparison of the time spent on each activity and the result from the developed algorithm in this work and the smartphone application Equilab for test session 1.

	True	Developed Algorithm	Equilab
Stand	2 min 6 sec	2 min 21 sec	2 min 42 sec
Walk	22 min 32 sec	22 min 6 sec	21 min 30 sec
Trot	14 min 12 sec	14 min 55 sec	14 min 20 sec
Canter	9 min 28 sec	8 min 56 sec	9 min 46 sec

6.3.2 Result Test Session 2

The classification accuracy for test session 2 is 97.4 % and is presented in Figure 6.4. The errors are marked in Figure 6.5 and the true times spent on activity can be compared to the times of the developed algorithm and the times obtained from Equilab in Table 6.3.

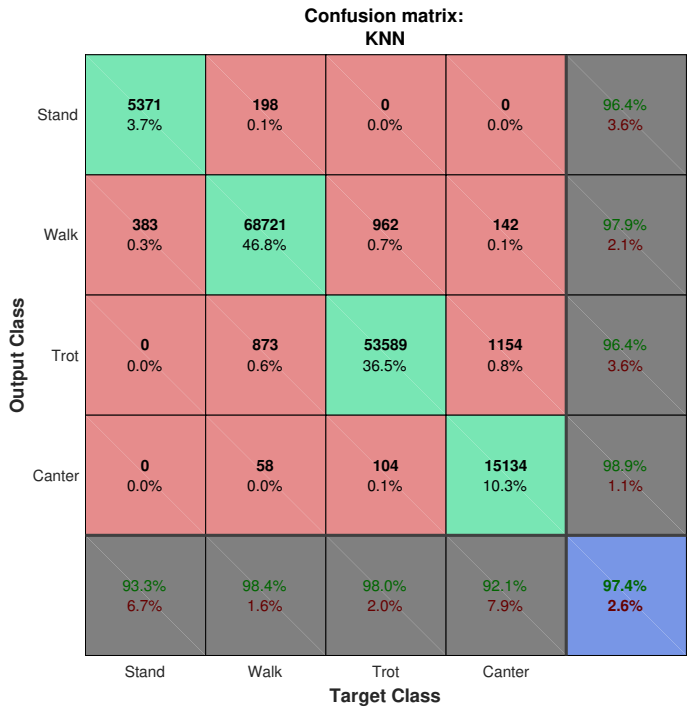

Figure 6.4: Confusion matrix of the result from test session 2. The overall classification accuracy is presented in the bottom right square.

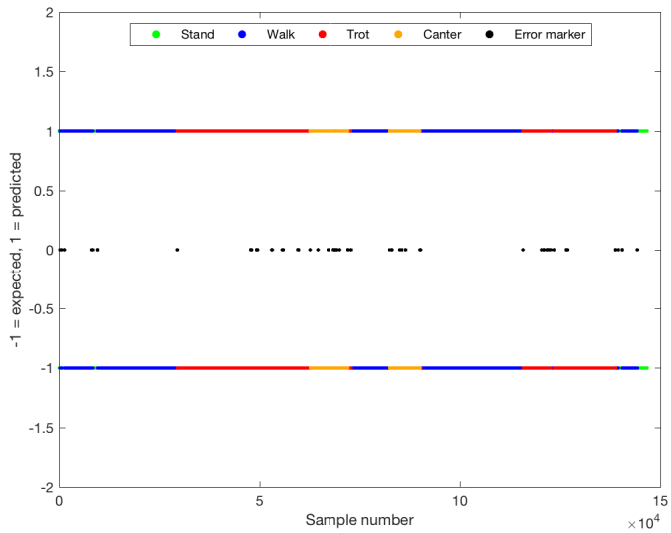

Figure 6.5: The plot visualizes the comparison of the expected (the line at -1) and the predicted (the line at 1) result from test session 2. The black marks (line at 0) visualize the errors. Note that the error marks are approximately 3 % of the expected and predicted lines.

Table 6.3: Comparison of the time spent on each activity and the result from the developed algorithm in this work and the smartphone application Equilab for test session 2.

	True	Developed Algorithm	Equilab
Stand	1 min 55 sec	1 min 51 sec	2 min 24 sec
Walk	23 min 17 sec	23 min 24 sec	22 min 48 sec
Trot	18 min 13 sec	18 min 32 sec	18 min 16 sec
Canter	5 min 28 sec	5 min 6 sec	5 min 25 sec

6.4 Computational Complexity

The developed algorithm in this work is fast when the classifiers have been trained. It should be feasible to run the developed algorithm in real time. A rough estimate of the computational complexity of the algorithm was done. The process of calculating the five needed features for a single window and classifying the window was tested on a i5-4200U CPU. With other applications running and the computer being in energy saving mode, i.e. capped clock speed (∼800 MHz), the feature calculation for all 5 features used and classification took approximately 0.03 seconds.

7. Discussion

In this chapter the data collection, the algorithm development and the final result of this master's thesis are discussed. Ethical aspects and future work are also described.

7.1 Data Collection

The data collection generally worked well and all collected data could be used in the development and the evaluation of the algorithm. The written protocol used during the collection was a good way to ensure that all the recorded riding sessions were performed in the same manner with sensor positions and labeling. The chosen sensor positions were easy to access. The phones could without difficulties be fasten and did not move during the riding sessions. However, the accelerometer of the phone fastened on the back of the rider did saturate. An accelerometer with a larger range would have been needed. One of the biggest issues was the labeling of data since it was done manually by writing down the labels with the help of a stopwatch. It was hard to capture all changes during a session especially when many rapid alternations between gaits happened during a short time period. Some riding sessions lasted a long time and it was hard to stay concentrated during the whole session. In the collection of the training data this was not a big problem since data that was potentially mislabeled or had irregular movement could be discarded. The collection of testing data was more critical since the developed algorithm was compared to the existing product Equilab. More controlled test session would probably have been helpful to ensure good labeling for the benchmark. The collected training set contains recordings from the same horse in two different sessions. This could possibly introduce some bias towards this horse, but since two different riders are used it captures the differences between them.

Overall the collected data set contained relatively good variance, but more data would strengthen the algorithm. Using the same methodology on a larger data set containing more variance would most likely change the final algorithm by adding other or more features or by changing classifier. The same phones were used in the same sensor position for all recordings. To reduce possible bias towards the phone sensors a larger variety of phones could be used. To capture more variance a larger variety of horses and riders would have been needed.

7.2 Algorithm Development

Many different features were implemented and used in the development of the algorithm. All of them were previously used in other works, but not all of them were used in combination. The final gait recognition algorithm contained three features from the time domain, one feature from the frequency domain and one wavelet based feature. Both the accelerometer and the gyroscope signals were represented in the final algorithm. When all feature sets selected by the different methods were studied it was noted that at least one feature from the accelerometer and one from the gyroscope were present. It was never examined if only one of the sensors would work

well on their own. From the result it seems like they work very well together, just as suggested in section 1.2.1. All sets did contain features from more than one domain.

The choice to use the smaller window length with 128 samples was due to the rapid changes observed during the recording sessions. One drawback with the smaller window size, compared to the larger window size with 256 samples, was that it had a harder time classifying stand.

The method sequential forward feature selection was used in the final algorithm. Regardless of the classifier used in the SFFS the different feature sets performed well for all the classifiers. The feature sets had some differences between them but they also contained shared features. One of the drawbacks with the method is that it does not necessarily find the best paired features due to the greedy nature of the algorithm. The first feature is the one that on its own gives the best classification. The first feature is then paired with a second feature that adds the most combined with the first selected feature. There is no guarantee that two other features does not provide a better combined classification. But the method is relatively fast and did, in this case, result in a feature set that seems to perform well. The best option would have been to test all combinations of features but since there were 405 implemented features in this work it would not have been feasible.

The time limit of this project and the size of the collected data set, limited the possibility to examine and study more classifiers. For example artificial neural networks and hidden Markov models were brought up in section 1.2.1 but was not studied deeper or examined during the thesis work. The possibility to have an additional class with unknown activity was discussed, but this was not further pursued, as the assumption was that the algorithm would only be used on gait data.

7.3 Classification

The results for the classification of the collected testing data indicate that the algorithm works well for equine gait recognition but further development and evaluation is needed. Most of the errors are found when the gait changes, as mentioned earlier, which probably is connected with the overlapping windows and the projection of the window label to a sample based label. If these errors could be reduced by the use of shorter windows or increased overlaps would be interesting to evaluate. The combination of a window length of 128 samples and 50 % overlap for 50 Hz sample rate indicates that the time resolution is good enough for the gait recognition purpose and thus no further evaluation of window lengths and overlaps were considered necessary.

Comparing the time spent on each activity with the times outputted by Equilab and by the developed algorithm, see Table 6.2 and 6.3, it can be seen that the times match well. It should be noted that Equilab was updated during the work of this master's thesis and the version released 24 April was used in the comparison. If problems with misclassified gaits (as mentioned in section 1.1) remains in the updated version of Equilab needs to be further evaluated since it could not be seen in this project. Some difference can be seen between Equilab and the developed

algorithm. For the developed algorithm, the total time for 'Stand' in Table 6.3 is closer to the true time than Equilab. Equilab does not present any time spent standing still in their application interface but when pressing the bottom "Details" in the app it can be seen that the riding is classified as either walk, trot, canter or stand. The application displays a total time for the recording session and the times spent walking, trotting and cantering. The value presented for the activity 'Stand' was thereby calculated by taking the total recording time minus the suggested time for the other activities. (In an updated version of the app in June 2017 they have replaced 'Stand' to 'Mix' instead, which would include standing still and activities not recognized. However, this was not taken into account since the results were compiled before the update was made and were based on the information in the previous version.) It seems like the developed algorithm in this work is better at classifying 'Stand' and 'Walk' while Equilab performs better for 'Trot' and 'Canter'. It should be noted that the labeling has been done manually and may contain human errors. The small deviations in time could thereby be due to the labeling. This results in the opinion that the developed algorithm in this project, as well as Equilab's unknown algorithm, performed very well on the test subjects.

A big advantage for Equilab is that the application is orientation independent and the phone running the algorithm could be placed in an optional pocket on the rider. Equilab also has an algorithm that runs in real time.

7.4 Ethics

This work is of interest since it is important for the rider to know if the horse has got enough exercise and if the exercise has been varying. The horse is not harmed during the collection of data since the horse is used to wear a lot of equipment and a smartphone with the chosen placement does not affect the horse. The riders were asked before the data was collected and none was forced to participate.

7.5 Future Work

To improve the algorithm further several things, additional to collection of more data with even larger variance, can be done. One improvement would be to include more breeds, e.g. ponies and Icelandic horses, since their motion pattern may be different compared to the horses that have been used in this project. Since Icelandic horses have additional gaits, e.g. tölt, additional gaits would have to be implemented to the algorithm.

Collecting data from more diverse grounds would also improve the algorithm since all the collected data was from medium soft ground. It would have been desirable to get data from harder grounds, for example riding in the woods and on asphalt.

Data was recorded from two different sensor positions. The idea was that a combination of the two sensor positions would strengthen the algorithm and also make it possible to construct a measurement of quality of the riding pattern. The com-

bination of both sensor positions was never evaluated due to time limitations but would be interesting to evaluate in future work. The quality measurement would need more research of horseback riding and help from experts to create a data set containing the necessary information.

Additional improvement would be to use another sensor which would be easy to start and stop with an external device. The external device could also make it possible to create flags for different events in the collected signal. This would make the labeling easier.

8. Conclusion

This chapter describes the main conclusions of this master's thesis.

In this master's thesis a training data set and a testing data set were collected. The training set contained five riding sessions including four horses and four riders. The testing set contained two riding session including two horses and two riders. The collections were made by following a proprietary protocol. An equine gait recognition algorithm based on accelerometer and gyroscope signals was developed. Conclusions were made to use data collected at the saddle-girth and a window length of 128 samples (2.56 seconds) with 50 % overlap in the final algorithm. Five features were included in the algorithm and two classifiers using two respectively three of the features. The first classifier separated stand and gaits. The second classifier classified gaits as either walk, trot or canter. The classifiers used in both cases was KNN with K = 3.

It was concluded that the algorithm performed well on the collected testing data set with an accuracy of 94 % and 97 %, respectively. It should be noted that the same phone used for developing the algorithm was used when collecting the testing data. The performance of the developed algorithm on the test set was benchmarked against the product Equilab and it was concluded that the performance of both algorithms were similar.

Further development of the algorithm will be needed to include other types of terrains and a larger variety of horses and riders.

Bibliography

[1] S. Miesner, M. Putz, M. Plewa, and A. Frömming, *Ridhandboken 1 grundutbildning för ryttare och häst*, 5th ed. Svenska Ridsportsförbundet, 2011, ISBN: 978-91-631-4472-1.

[2] RSPCA Australia Knowledgebase, *How much should I exercise my horse?* Mar. 9, 2016. [Online]. Available: `http://kb.rspca.org.au/How-much-should-I-exercise-my-horse_467.html` (visited on 06/05/2017).

[3] Hippson, *Rid-appen equilab blev årets pryl 2016!* Feb. 16, 2017. [Online]. Available: `https://www.hippson.se/artikelarkivet/prylkollen/rid-appen-equilab-blev-arets-pryl.htm` (visited on 06/05/2017).

[4] T. Schlömer, B. Poppinga, N. Henze, and S. Boll, "Gesture recognition with a Wii controller," in *Proceedings of the 2nd international conference on Tangible and embedded interaction - TEI*, Association for Computing Machinery (ACM), 2008. DOI: `10.1145/1347390.1347395`.

[5] J. A. V. Diosdado, Z. E. Barker, H. R. Hodges, J. R. Amory, D. P. Croft, N. J. Bell, and E. A. Codling, "Classification of behaviour in housed dairy cows using an accelerometer-based activity monitoring system," *Animal Biotelemetry*, vol. 3, no. 1, Jun. 2015. DOI: `10.1186/s40317-015-0045-8`.

[6] P. Martiskainen, M. Järvinen, J.-P. Skön, J. Tiirikainen, M. Kolehmainen, and J. Mononen, "Cow behaviour pattern recognition using a three-dimensional accelerometer and support vector machines," *Applied Animal Behaviour Science*, vol. 119, no. 1-2, pp. 32–38, Jun. 2009. DOI: `10.1016/j.applanim.2009.03.005`.

[7] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *Lecture Notes in Computer Science*, Springer Nature, 2004, pp. 1–17. DOI: `10.1007/978-3-540-24646-6_1`.

[8] A. Mannini and A. M. Sabatini, "Machine learning methods for classifying human physical activity from on-body accelerometers," *Sensors*, vol. 10, no. 2, pp. 1154–1175, Feb. 2010. DOI: `10.3390/s100201154`.

[9] C. A. Ronao and S.-B. Cho, "Recognizing human activities from smartphone sensors using hierarchical continuous hidden markov models," *International Journal of Distributed Sensor Networks*, vol. 13, no. 1, Jan. 2017. DOI: `10.1177/1550147716683687`.

[10] M. Shoaib, S. Bosch, O. Incel, H. Scholten, and P. Havinga, "Complex human activity recognition using smartphone and wrist-worn motion sensors," *Sensors*, vol. 16, no. 4, Mar. 2016. DOI: `10.3390/s16040426`.

[11] J. Margarito, R. Helaoui, A. Bianchi, F. Sartor, and A. Bonomi, "User-independent recognition of sports activities from a single wrist-worn accelerometer: A template matching based approach," *IEEE Transactions on Biomedical Engineering*, pp. 788–796, 2015. DOI: `10.1109/tbme.2015.2471094`.

[12] B. Ayrulu-Erdem and B. Barshan, "Leg motion classification with artificial neural networks using wavelet-based features of gyroscope signals," *Sensors*, vol. 11, no. 12, pp. 1721–1743, Jan. 2011. DOI: `10.3390/s110201721`.

[13] J.-B. Burla, A. Ostertag, H. S. Westerath, and E. Hillmann, "Gait determination and activity measurement in horses using an accelerometer," *Computers and Electronics in Agriculture*, vol. 102, pp. 127–133, Mar. 2014. DOI: 10.1016/j.compag.2014.01.001.

[14] J. J. Robilliard, T. Pfau, and A. M. Wilson, "Gait characterisation and classification in horses," *Journal of Experimental Biology*, vol. 210, no. 2, pp. 187–197, Jan. 2007. DOI: 10.1242/jeb.02611.

[15] E. Barrey and P. Galloux, "Analysis of the equine jumping technique by accelerometry," *Equine Veterinary Journal*, vol. 29, no. S23, pp. 45–49, Jun. 2010. DOI: 10.1111/j.2042-3306.1997.tb05052.x.

[16] S. Viry, R. Sleimen-Malkoun, J.-J. Temprado, J.-P. Frances, E. Berton, M. Laurent, and C. Nicol, "Patterns of horse-rider coordination during endurance race: A dynamical system approach," *PLoS ONE*, vol. 8, no. 8, e71804, Aug. 2013. DOI: 10.1371/journal.pone.0071804.

[17] Equilab, 2017. [Online]. Available: https://equilab.horse/ (visited on 06/05/2017).

[18] Google Play, *Equilab - GPS hästträning*, Apr. 24, 2017. [Online]. Available: https://play.google.com/store/apps/details?id=horse.schvung.equilab (visited on 06/05/2017).

[19] Equisense, 2016. [Online]. Available: http://www.equisense.com/en/ (visited on 06/05/2017).

[20] Protequus LLT, *Nightwatch*, 2017. [Online]. Available: www.nightwatch24.se (visited on 05/12/2017).

[21] M. J. McCracken, J. Kramer, K. G. Keegan, M. Lopes, D. A. Wilson, S. K. Reed, A. LaCarrubba, and M. Rasch, "Comparison of an inertial sensor system of lameness quantification with subjective lameness evaluation," *Equine Veterinary Journal*, vol. 44, no. 6, pp. 652–656, May 2012. DOI: 10.1111/j.2042-3306.2012.00571.x.

[22] M. Shoaib, H. Scholten, and P. Havinga, "Towards physical activity recognition using smartphone sensors," in *2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing*, Institute of Electrical and Electronics Engineers (IEEE), Dec. 2013. DOI: 10.1109/uic-atc.2013.43.

[23] D. Smith, *How does an accelerometer work in a smartphone? Bill Hammack, the engineer guy explains*, May 23, 2012. [Online]. Available: http://www.ibtimes.com/how-does-accelerometer-work-smartphone-bill-hammack-engineer-guy-explains-full-text-699762 (visited on 06/05/2017).

[24] V. Kempe, *Inertial mems: Principles and practice*. Cambridge Univ Pr, Feb. 11, 2011, ch. 8, ISBN: 0521766583. [Online]. Available: http://www.ebook.de/de/product/13051458/volker_kempe_inertial_mems_principles_and_practice.html.

[25] S. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Pattern Anal. and Machine Intell.*, vol. 11, no. 7, pp. 674–693, 1989. DOI: 10.1109/34.192463.

[26] C. M. Bishop, *Pattern recognition and machine learning*. Springer-Verlag New York Inc., Aug. 17, 2006, ISBN: 0387310738. [Online]. Available: http://www.ebook.de/de/product/5324937/christopher_m_bishop_pattern_recognition_and_machine_learning.html.

[27] Mathworks, *Fit binary classification decision tree for multiclass classification*. [Online]. Available: https://se.mathworks.com/help/stats/fitctree.html (visited on 06/05/2017).

[28] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*. Springer-Verlag GmbH, Jul. 11, 2014, ch. 8, ISBN: 1461471370. [Online]. Available: http://www.ebook.de/de/product/20292548/gareth_james_daniela_witten_trevor_hastie_robert_tibshirani_an_introduction_to_statistical_learning.html.

[29] T. Hastie, R. Tibshirani, and J. Friedman, *Elements of statistical learning*. Springer-Verlag New York Inc., Feb. 9, 2009, ch. 15, 745 pp., ISBN: 0387848576. [Online]. Available: http://www.ebook.de/de/product/8023140/trevor_hastie_robert_tibshirani_jerome_friedman_elements_of_statistical_learning.html.

[30] L. Breiman, "Random forest," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. DOI: 10.1023/a:1010933404324.

[31] Y. Ma and G. Guo, *Support vector machines applications*. Springer, Mar. 3, 2014, pp. 23–26, ISBN: 3319022997. [Online]. Available: http://www.ebook.de/de/product/21217233/support_vector_machines_applications.html.

[32] Mathworks, *Train binary support vector machine classifier*. [Online]. Available: https://se.mathworks.com/help/stats/fitcsvm.html (visited on 06/05/2017).

[33] A. R. Webb, K. D. Copsey, and G. Cawley, *Statistical pattern recognition*, 3rd ed. John Wiley and Sons Ltd, Oct. 21, 2011, pp. 454–455, ISBN: 0470682272. [Online]. Available: http://www.ebook.de/de/product/13921802/andrew_r_webb_keith_derek_copsey_gavin_cawley_statistical_pattern_recognition.html.

[34] O. Tunçel, K. Altun, and B. Barshan, "Classifying human leg motions with uniaxial piezoelectric gyroscopes," *Sensors*, vol. 9, pp. 8508–8546, Sep. 27, 2009.

[35] L. I. Smith *et al.*, "A tutorial on principal components analysis," *Cornell University, USA*, vol. 51, no. 52, p. 65, 2002.

[36] Google Play, *Sensor Kinetics*, Jun. 5, 2015. [Online]. Available: https://play.google.com/store/apps/details?id=com.innoventions.sensorkinetics (visited on 06/05/2017).

[37] Google Play, *AndroSensor*, Jan. 23, 2015. [Online]. Available: https://play.google.com/store/apps/details?id=com.fivasim.androsensor (visited on 06/05/2017).