RUPRECHT-KARLS-UNIVERSITÄT HEIDELBERG

HOCHSCHULE HEILBRONN

Medizinische Informatik

Human Activity Pattern Recognition from Accelerometry Data

Master's Thesis

written and presented by

Dipl.-Ing. (FH) Dennis Jos

in fulfillment of the requirements for the academic degree of

Master of Science

completed at the

German Aerospace Center

Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)

Institute of Aerospace Medicine

Department of Space Physiology

Cologne, Germany

supervised by

Prof. Dr. Oliver Kalthoff, Hochschule Heilbronn

Dr. rer. nat. Uwe Mittag, DLR

Cologne, November 2013

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

HOCHSCHULE HEILBRONN

DLR Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center

Abstract English

Title of master's thesis
Human activity pattern recognition from accelerometry data

Author
Dipl.-Ing. (FH) Dennis Jos

First examiner
Prof. Dr. Oliver Kalthoff

Second examiner
Dr. rer. nat. Uwe Mittag

Ambulant studies are dependent on the behavior and compliance of subjects in their home environment. Especially during interventions on the musculoskeletal system, monitoring physical activity is essential, even for research on nutritional, metabolic, or neuromuscular issues. To support an ambulant study at the *German Aerospace Center* (*DLR*), a pattern recognition system for human activity was developed. Everyday activities of static (standing, sitting, lying) and dynamic nature (walking, ascending stairs, descending stairs, jogging) were under consideration. Two tri-axial accelerometers were attached to the hip and parallel to the tibia. Pattern characterizing features from the time domain (mean, standard deviation, absolute maximum) and the frequency domain (main frequencies, spectral entropy, autoregressive coefficients, signal magnitude area) were extracted. Artificial neural networks (ANN) with a feedforward topology were trained with backpropagation as supervised learning algorithm. An evaluation of the resulting classifier was conducted with 14 subjects completing an activity protocol and a free chosen course of activities. An individual ANN was trained for each subject. Accuracies of 87,99 % and 71,23 % were approached in classifying the activity protocol and the free run, respectively. Reliabilities of 96,49 % and 76,77 % were measured. These performance parameters represent a working ambulant physical activity monitoring system.

Key words
Activity recognition, accelerometer, artificial neural networks, ambulatory monitoring, supervised learning

Abstract Deutsch

Thema der Masterarbeit
Mustererkennung menschlicher Bewegungen in Beschleunigungsdaten

Autor
Dipl.-Ing. (FH) Dennis Jos

Erstprüfer
Prof. Dr. Oliver Kalthoff

Zweitprüfer
Dr. rer. nat. Uwe Mittag

Bei ambulanten Studien ist das Mitwirken des Probanden (Compliance) von großer Bedeutung. Besonders bei Interventionsstudien, die den menschlichen Bewegungsapparat untersuchen, ist es wichtig, die alltäglichen Aktivitäten in heimischer Umgebung zu protokollieren. Körperliche Aktivität hat Einfluss auf Messungen, die Auswirkungen von Ernährung, Stoffwechsel oder neuromuskulären Stimulationen quantifizieren. Zur Unterstützung einer ambulanten Studie am *Deutschen Zentrum für Luft- und Raumfahrt* (*DLR*) wurde ein System zur Erkennung menschlicher Bewegungen entwickelt. Alltägliche Aktivitäten von statischer (Stehen, Sitzen, Laufen) und dynamischer Natur (Gehen, Treppen auf- bzw. absteigen, Joggen) wurden untersucht. Zwei dreiachsige Beschleunigungssensoren wurden an der Hüfte und parallel zum Schienbein montiert. Von den Beschleunigungssignalen wurden musterbeschreibende Merkmale aus dem Zeitbereich (Mittelwert, Standardabweichung, Betragsmaximum) und Frequenzbereich (Hauptfrequenzen, spektrale Entropie, autoregressive Koeffizienten, Fläche der Signalamplituden) extrahiert. Künstliche neuronale Netze (ANN) mit einer Feedforward-Struktur wurden mit Backpropagation und überwachtem Lernen trainiert. Für die Evaluation des daraus entstandenen Klassifikators absolvierten 14 Testpersonen einen Aktivitätsparcours und einen freien Lauf mit beliebiger Reihenfolge von Aktivitäten. Für jeden Probanden wurde ein individuelles ANN trainiert. Genauigkeiten von 87,99 % und 71,23 % wurden bei der Klassifikation des Aktivitätsparcours bzw. des freien Laufs gemessen. Die Zuverlässigkeit der Klassifizierungen lag bei 96,49 % bzw. 76,77 %. Diese Leistungsparameter beschreiben ein funktionierendes ambulantes Monitoring System von körperlichen Aktivitäten.

Schlüsselwörter
Bewegungserkennung, Beschleunigungssensoren, künstliche neuronale Netze, ambulantes Monitoring, überwachtes Lernen

Eidesstattliche Erklärung

Universität Heidelberg
Hochschule Heilbronn
Studiengang Medizinische Informatik

Autor	***Matrikelnummer***
Dipl.-Ing. (FH) Dennis Jos	179715

Thema der Masterarbeit
Human Activity Pattern Recognition from Accelerometry Data

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt und ist auch noch nicht veröffentlicht.

Köln, den 26. November 2013

_____
Dennis Jos

Acknowledgement

This master's thesis was carried out at the *German Aerospace Center* in Cologne. Support was given by the *Institute of Aerospace Medicine*, who had funded this work in all its stages. I gratefully acknowledge the collaboration and help provided by Dr. Uwe Mittag as supervisor and Kathrin Schopen as conducting researcher of the *NutriHEP* study throughout the whole thesis. Further, I would like to thank Prof. Dr. Oliver Kalthoff for his supervision. Thanks are also due to my colleagues at the *Department of Space Physiology*, who volunteered as subjects during the evaluation study. I also thank Michel Ducos and Prof. Dr. Jörn Rittweger for their constructive comments in the early stages of this work. Valuable advice in scripting this thesis is gratefully acknowledged to Josefine Kupper and Miriam Reisenhofer.

List of contents

List of figures

List of tables

List of abbreviations

AC	Activity course
ANN	Artificial neural network
AR	Autoregressive
CD	Compact disc
CSV	Comma separated values
DAT	Data file
DFT	Discrete Fourier Transform
DLR	*Deutsches Zentrum für Luft- und Raumfahrt* (*German Aerospace Center*)
FFT	Fast Fourier Transform
FFTW	*Fast Fourier Transform in the West*
FR	Free run
GCDC	*Gulf Coast Data Concepts*
GUI	Graphical user interface
HEP	*HEPHAISTOS* study
ID	Identification
MCR	*MATLAB Compiler Runtime*
MEMS	Micro-electromechanical system
NASA	*National Aeronautics and Space Administration*
NutriHEP	Nutrition study using HEP orthosis
PSD	Power spectral density
Req	Requirement
SD	Standard deviation
SE	Spectral entropy
SMA	Signal magnitude area
SysReq	System requirement
TXT	Text file
USB	Universal Serial Bus

1. Introduction

1.1. Motivation

With respect to future long duration space missions to Mars or beyond, muscle and bone loss are a major concern at the astronaut's health. Since these processes have been studied insufficiently, there are no satisfying therapies or precautions available yet. Consequently, counter actions are required to restore health and physical capability of astronauts during long duration space missions. Research in this field is progressing with immobilizing studies on earth, where the loading of muscles and bones is reduced for a period of time. This unloading intervention can be applied either with bed rest to the whole body or partly with a custom lower leg orthosis. Conditions and metabolisms in muscles and bones are monitored before, during, and after the intervention. Whereas during bed rest studies the subject is not allowed to get out of bed, an orthosis study leaves the opportunity for the subject to move freely, because an ambulant setting is feasible. Latter aspect involves that the daily physical activity of the subject is unknown or can only be protocoled by the subject herself/himself. Since physical activity has an influence on muscle and bone condition, there is a need to monitor the subject's daily physical activity.

This master's thesis is conducted on behalf of the *German Aerospace Center* (*DLR*) in Cologne, Germany. In 2014, the *Department of Space Physiology* at the *Institute of Aerospace Medicine* will start the *NutriHEP* study. This study investigates the influence of nutrition and neuromuscular stimulation on the local insulin sensitivity in the calf muscle during immobilization. Subjects are partly immobilized by wearing the custom-built *HEPHAISTOS* (*HEP*) orthosis on one leg. This orthosis, developed by WEBER ET AL. [WDM+13], unloads the muscles of the lower limb while retaining the application of body weight on the skeletal structure. It is designed for ambulant studies and allows everyday locomotion. This directed immobilization of the calf muscle induces muscle atrophy which affects insulin sensitivity. In the *NutriHEP* study, electrical stimuli and a specific diet with lupine seeds will be tested for their effectiveness on insulin sensitivity. Any influence on insulin sensitivity may affect the degree of muscle loss. It is required to validate the relationship between muscle loss and orthosis intervention. A possible side effect of wearing the orthosis may be that the subject is less active. This whole body inactivity can call systemic effects on the insulin sensitivity of the muscles [WSW+13]. To locate interventional effects on the calf muscle only, it is essential to show that the physical activity of the whole body remains the same. Activity logging by the subject is neither reliable nor accurate. Therefore, an ambulant monitor-

ing system of the physical activity is needed. Furthermore, a documentation of the subject's compliance is required to ensure that the orthosis was worn throughout the study.

For an ambulant study, accelerometers are a low-cost and practical solution to monitor physical activity. Tri-axial accelerometers are attached to the orthosis and the subject's hip to acquire data of acceleration. These data sets are used to assess different daily physical activities, since *"accelerometry is preferred because acceleration is proportional to external force and hence can reflect intensity and frequency of human movement"* [YaHs10]. Classification methods are then applied to recognize patterns of human activity in the acceleration data. Several solutions with different methods for this process do exist, but no gold standard can be identified. Therefore, one aim of this thesis is to find a new combination of methods for activity recognition. The other aim is to develop of a human activity recognition software program which will be ready to operate during the *NutriHEP* study.

In conclusion, this master's thesis focuses on the definition and implementation of an analyzing method which uses accelerometry data to identify, characterize and allocate acceleration patterns to human physical activity.

1.2. Thesis outline

In chapter 2 the user requirements are acquired and derived into system requirements. Chapter 3 is dedicated to explaining the materials and methods used in this thesis and summarizing the state of research. Chapter 4 deals with decisions towards the architectural design and the data flow is visualized and explained in detail. In chapter 5, the implementation of the concept into an operable software program is described and the major functions and their source codes are covered. The evaluation of the software program follows in chapter 6. Its results and the course of this thesis are discussed in a final conclusion in chapter 7.

2. Requirements

2.1. User requirements

The aim of this thesis is to develop a software program which will be used during the *NutriHEP* study. The conducting researcher of this study will be also the end user of the software. Therefore, this person was interviewed to identify what her requirements (Req) and expectations were. The following questions and statements of the researcher imply the main aspects which were mentioned.

Req1 "When did the subject do which body movement?"

The analysis software should have the ability to tell when and how the subject moved wearing the *HEP* orthosis. Figure 2-1 shows an example of a daily overview. It was created in collaboration with the conducting researcher. The diagram shows activity as a function of time. Figure 2-1 is an example of a possible outcome of the software.

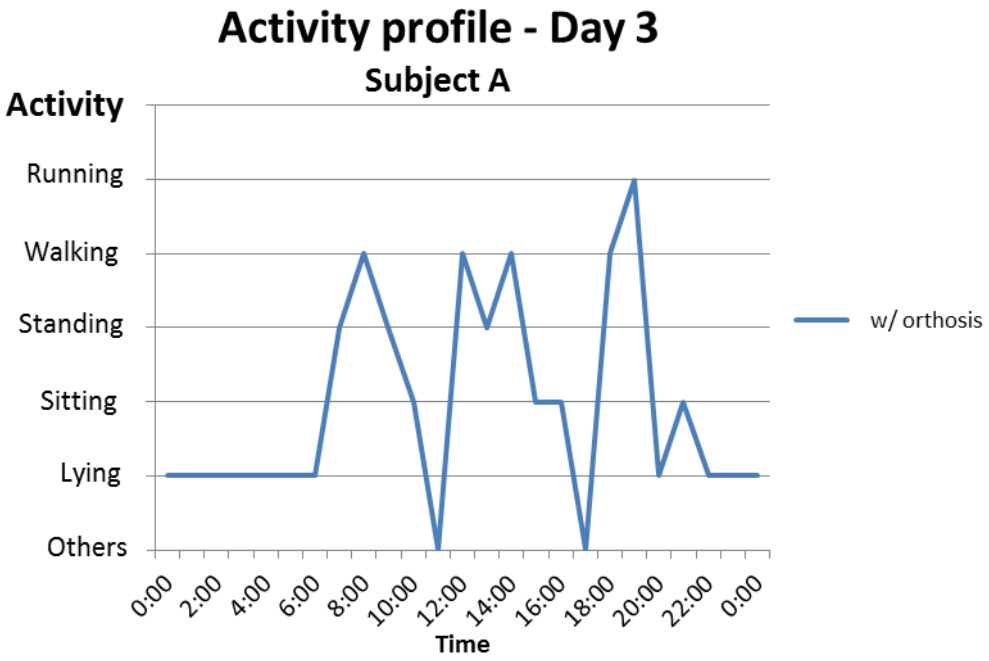


Figure 2-1: Example of graphical visualization for Req1

Req2 "How long did the subject do which body movement?"

The analysis software should have the ability to tell how long the subject moved in a certain way. According to the interviewee, a representation in the form of a pie chart would be helpful, because it provides a proper overview of the relative distribution of the different activities

Req3 "Was the orthosis worn the entire time?"

The analysis software should have the ability to tell when the orthosis was worn or not worn by the subject. Figure 2-2 shows an example of a weekly overview, which was created in collaboration with the conducting researcher. In this graphic, the compliance is binary color coded. It is easy to detect time intervals where the orthosis was not worn for a quick interrogation with the subject. Furthermore, this requirement implies that the software should be applicable with the usage of an orthosis.

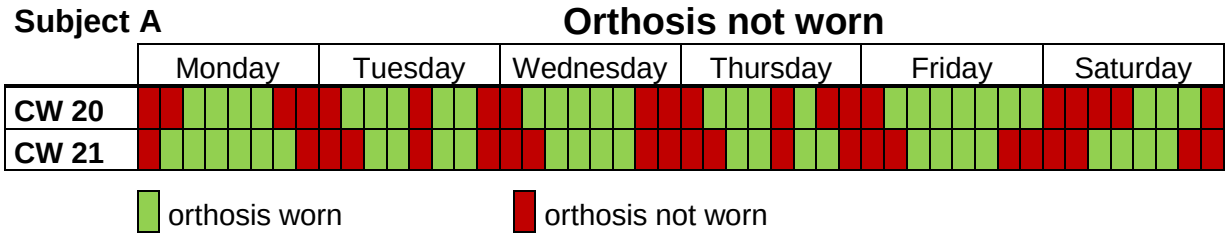


Figure 2-2: Example of graphical visualization for Req3

Req4 "What does the data look like during unidentified activities?"

The analysis software should have the ability to show raw acceleration data collected during unidentified activities for further manual investigation.

Req5 "The software should be easy to use."

Interviewing the conducting researcher of the *NutriHEP* study it turned out that the computational skills of potential users may bear insecurities. It was stressed that the handling of the software should be straightforward and intuitive. In addition, the researcher proposed that there might be a lack of time during the data analysis of the study. Therefore, a colleague with no experience in data analysis should be able to work with the software.

Req6 "The software should work as a standalone executable."

The preliminary requirement implies that several persons will be working with the analysis software. Hence, the software should run on different work stations. A standalone executable of the analysis software ensures parallel work on independent computers.

Req7 "The software should be provided in English and German."

The main language of the software should be English. However, it is possible that potential users, as stated in Req5, do not speak any English. Thus, the user should be offered to choose between English and German version of the software program.

Req8 "The software should not be time consuming."

The analysis software should have as many automatic procedures as possible. It should not be too time consuming, because it is a supporting process of the *NutriHEP* study.

Req9 "The software should work with a single sensor on any position."

The analysis software should be usable beyond the *NutriHEP* study. Whereas *NutriHEP* supplies the subject with two accelerometers (i.e. tibia and hip), the software should also work with a single sensor and at any chosen body position.

Req10 "The software should work with GCDC accelerometers."

GCDC accelerometers are sensors manufactured by *Gulf Coast Data Concepts* (Waveland, MS, USA). These accelerometers were used during the *HEP* study, a previous study run to develop the *HEP* orthosis. The *NutriHEP* study will be working with these accelerometers as well. Therefore, the concept of the analysis software should be focused on the processing of data acquired with *GCDC* accelerometers. In addition, the software might offer the possibility for a post-hoc analysis of the *HEP* data.

2.2. System requirements

Interpreting the previous user requirements, various system requirements (SysReq) for the development of this software program arise. The analysis of acceleration data requires numerous procedures and methods.

Interpreting Req1 and Req2

Concluding from acceleration data to body movement is a common classification problem. It needs to be identified to which category a new observation belongs. From a plethora of classification methods available, the following ones occur in many publications: Bayes classifier [LoYA09], decision tree classifier [LoYA09], and artificial neural networks [HGP+11] [KLLK10]. Further explanation of these methods will be given in chapter 3.3.

Classification algorithms cannot process raw signal data. Instead, they find correlations and similarities in features. Features are properties of raw signals which allow representing the characteristics of a signal by fewer values. Consequently, a feature extraction of the acceleration data is needed to make it identifiable and classifiable. Numerous methods of feature extraction are available. Algorithms derived from descriptive statistics [HGP+11] [DSD+12] [GeMC06] and the Fourier analysis [HGP+11] [KLLK10] are mostly used. Further explanation of these methods will be given in chapter 3.2.

The raw signal data from the accelerometers are not scaled in units of acceleration, but in raw count data. Moreover, the signal data may contain noise or other interferences. To this end, the raw data has to run through another procedure before feature extraction. The conversion into units of acceleration is dependent on the accelerometer settings. Several filtering methods are used for noise reduction: moving average [KLLK10], low-pass filter [HGP+11] [GeMX06] [LoYA09], and band-pass filter [YaHs10]. Further explanation of these methods will be given in chapter 3.1.

An evaluation of the classifying system is necessary to verify the accuracy of the software program. Since no requirement for minimum accuracy was formulated, a literature research had to be performed. Research showed that classification accuracy varies from 80 % [KNEK08] [LoYA09] [LKK+10] to over 90 % [PGKH09] [KwWM11] [KLLK10] [KhLK08] [HGP+11]. More detailed information of other literature will be covered in chapter 3.6. The required user's accuracy is thus set to 80 % for the software program.

A numerical computing environment is needed to implement these procedures, preferably one with built-in functions for signal processing and the chosen classification method.

Interpreting Req3

The stated requirement is a two-class classification problem. The classification methods described above may also be applicable in this scenario. Finding a simple decision rule, which only distinguishes between compliance and non-compliance, is however less complex. Further explanation of this decision process will be given in chapter 4.2.4.

Interpreting Req4 to Req9

Requirements Req4 to Req9 are directed to the user interface and the application of the software program and have no effect on the analyzing algorithm. Usage related requirements (Req4, Req5, Req7, and Req8) are to be concerned when designing the graphical user interface (GUI). Compiling software is necessary to compile a standalone executable (Req6).

Interpreting Req10

Like with most mechanical sensors, accelerometers measure the impact they are exposed to by changing voltage. These changes need to be converted into acceleration data using specific calibration equations, which are to be investigated. Further explanation of the conversion process will be given in chapter 3.4.

In conclusion, the following system requirements arise from the user requirements:

SysReq1 Classification method for pattern recognition
SysReq2 Feature extraction methods for signal description
SysReq3 Signal processing methods for pre-processing
SysReq4 Evaluation method for determining user's accuracy
SysReq5 Activity classification with more than 80 % user's accuracy
SysReq6 Numerical computing environment for programming
SysReq7 Decision rule for checking compliance
SysReq8 User-friendly graphical interface
SysReq9 Compiling software for generating standalone executables

3. Materials and methods

Analysis of acceleration data needs various procedures and methods. First, the raw data must obtain a pre-processing. Subsequently, the resulting signal is characterized by certain features. Based on these combined feature patterns, different activities can be identified by a classification method. Finally, the classifying system is tested in an evaluation. According to these required procedures, several methods for pre-processing, feature extraction, classification and evaluation are under consideration. This chapter explains possible methods for further analysis.

3.1. Pre-processing methods

Filtering

Digital filters are used for separating signals that have been combined and for restoring distorted signals. Analog filters are applicable for the same tasks, but digital filters achieve better results. [Smit03] Moving average filters, as they are used in [KLLK10], are mostly incorporated to filter out random noise. Each point in the output signal is produced by averaging a number of points from the input signal. Incrementing the averaging number leads to a decreased noise in the signal. However, the step response is lowered. It is mainly used for its smoothing effect. Low-pass filters process an input signal by passing only low-frequency components. They reduce the amplitude of high-frequency signals above a specific cutoff-frequency. Hence, low-pass filters are used to reduce high-frequency noise. Band-pass filters pass frequencies in a certain range. Amplitudes of signals with frequencies outside this range will be reduced. They can be incorporated by combining a low-pass and high-pass filter simultaneously. Therefore, a band-pass filter has two cutoff-frequencies.

The filtering methods above are the basis of signal separation and restoration and have been applied by HANSON ET AL. [HGP+11], KHAN ET AL. [KLLK10], LONG ET AL. [LoYA09] and many more. Numerous implementations of different filtering methods are available and easy to include, but due to the manipulating effect, the possibility of losing important information from the signal is present.

Moving Window

To extract several features from a signal stream, the data needs to be divided by small windows. This ensures an efficient computation time, considering the idea of the divide-and-conquer paradigm. Additionally, it is inevitable for the classification of activities. A window, which contains a certain number of data samples, can be identified as one activity. Therefore, the window size corresponds to the resolution of activity classification. For example, a window with a length of 10 seconds allows the classification of an activity every 10 seconds. This is the case, if the next window is consecutive. The resolution can be increased by using overlapping moving windows.

3.2. Feature extraction methods

During the process of feature extraction, several parameters are calculated from the acceleration signal, representing the characteristics of an activity, and thus making them identifiable. This section illustrates their basics, taken from the fields of descriptive statistics and Fourier analysis.

3.2.1. Descriptive statistics

Mean value

The mean value describes the average value of a digital signal. Measuring with tri-axial accelerometers, the individual means of all three axes in combination represent the orientation of the sensor towards the gravitational field of the earth. It is easy to calculate and an important indicator for body orientation.

Standard deviation

The standard deviation (SD) is a measure of dispersion from the signal's mean value. It appears plausible that more fluctuation of the signal corresponds with more activity of the sensor, and consequently of the subject. It can serve as indicator for the dynamic nature of a signal. The method for classifying the dynamic nature will be described in chapter 4.2.2.

Absolute maximum

The maximum value refers to the highest peak within a defined time window. A negative or positive acceleration determines the direction of acceleration along its measuring axis. Considering peaks in both directions, the maximum of the absolute values is taken.

Magnitude of acceleration vector

Acceleration is recorded using tri-axial accelerometers, which divide the experienced impact into measurements in three spatial axes. Since these axes are linear independent, they can be concatenated to a three-dimensional vector. The length of this vector represents the efficient value of the resulting acceleration and is calculated with the Euclidian norm. This absolute acceleration is mainly used for easier understanding in graphical visualizations of acceleration. Classifying the dynamic nature of a signal in chapter 4.2.2 will utilize this descriptive feature as well.

The descriptive methods above are considered as *"traditional features that are used for acceleration activity recognition"* [GjGC10] and have been applied by HANSON ET AL. [HGP+11], PREECE ET AL. [PGKH09], BOUTEN ET AL. [BKV+97] and many more.

3.2.2. Fourier analysis

Human daily activities, such as walking, running or climbing stairs, have periodic movement patterns. This suggests transforming the signal into the frequency domain and investigating its spectral components with a Fast Fourier Transform (FFT). In this representation, the main periods are represented by non-zero values at the corresponding frequency axis value [FDFC10]. It has been shown that different physical activities contain different dominant main frequencies [HGP+11] [PGKH09] [FCFD10].

Fast Fourier Transform

The Fourier Transform is the link between representing a signal in its time domain or in its frequency domain. In the time domain the signal is represented with a time dependency, the frequency domain shows a frequency dependency. Physical signals are often acquired in a time-discrete manner, resulting in equidistant sampled data. The Discrete Fourier Transform (DFT) is applicable for the numerical analysis of such signals. [Müll13]

NEUBAUER [Neub12] explains that the DFT allocates to a finite signal series in the time domain of length N with index $0 \leq t \leq N-1$

$$\{x(t)\}_{0 \leq t \leq N-1} = \{x(0), x(1), \dots, x(N-1)\}$$ Equation 3-1

adapted from [Neub12]

the finite spectral series in the frequency domain

$$\{X(f)\}_{0 \leq f \leq N-1} = \{X(0), X(1), \dots, X(N-1)\}$$ Equation 3-2

adapted from [Neub12]

consisting of N spectral values with index $0 \leq f \leq N-1$. The transformation rule is

$$X(f) = \sum_{t=0}^{N-1} x(t) \cdot e^{-j2\pi tf/N}.$$ Equation 3-3

adapted from [Neub12]

The DFT is written as

$$X(f) = DFT\{x(t)\}.$$ Equation 3-4

adapted from [Neub12]

The FFT is another way to calculate the Discrete Fourier transform. It approaches the same results like other methods, but is more efficient and less time-consuming. It can reduce the computation time of an input signal by hundreds. [Smit03]

A divide-and-conquer application leads to a decreasing number of required operations [Müll13]. According to the Radix-2 FFT algorithm by COOLEY AND TUKEY [CoTu65] several symmetric properties allow a numerical efficient calculation of the DFT [Neub12]. Implementations of the FFT allow signal analysis in the frequency domain without higher computational expense.

In *MATLAB* the FFT functions are based on the *FFTW* library [FrJo98] using the Cooley-Tukey algorithm [CoTu65], to compute an N-point DFT (N needs to be composite, i.e. $N = N_1 \cdot N_2$). First, N_1 transforms of size N_2 are computed, and then N_2 transforms of size N_1. The decomposition is applied recursively until the problem can be solved using one of several machine-generated fixed-size codes. These codes include combinations of the Cooley-Tukey algorithm [OpSc10], a prime factor algorithm [OpSc10], and a split-radix algorithm [DuVe90]. The computation time of the *MATLAB* FFT function depends on the length of the transform. [Math13]

The FFT is a basic tool in signal analysis and an essential part of activity recognition. It is written as

$$Y(f) = FFT\{x(t)\}.$$ Equation 3-5

adapted from [Math13a]

Periodogram

As stated in Equation 3-5, $Y(f)$ gives the distribution of the Fourier coefficients in the complex plane. The power of the input signal is the squared complex magnitude of $Y(f)$(see Equation 3-6). Plotting it against its scaled frequency (see Equation 3-7) shows the periodogram. [Math13a]

$$Pow(f) = \left|\left\{Y(0), Y(1), \dots, Y\left(\frac{N}{2} - 1\right)\right\}\right|^2$$

Equation 3-6
adapted from [Math13a]

$$Frq = \frac{\left\{0,1, \dots, \frac{N}{2} - 1\right\}}{(N)}$$

Equation 3-7
adapted from [Math13a]

Peak magnitude

Peak magnitudes are local maxima in the periodogram. Their frequencies represent the most dominant frequencies in the input signal. The strongest frequency Frq_{main} can be picked out by allocating the frequency with the highest power in the periodogram. [Math13a]

$$Pow(Frq_{main}) = max\big(Pow(f)\big)$$

Equation 3-8
adapted from [Math13a]

Power spectral density

The power spectral density (PSD) describes the average distribution of the signal's power in the frequency domain. The average power over a frequency band is computed by the integral of the PSD, and not its peaks [Math13b]. For time discrete signals, the PSD is calculated from the DFT of the autocorrelation sequence [Beuc11]. It is used for the calculation of the spectral entropy (SE).

Spectral entropy

According to information theory, entropy describes the sum of all microstates in a balanced system. A low number of different microstates appear in a consistent and uniform system with low entropy. In contrast, numerous variations of microstates cause uncertainty and confusion, hence, high entropy. Entropy is defined as a measurement of uncertainty or complexness of a system. [Alve07] [ShWe64]

For the differentiation between deterministic and random parts of a signal, entropy is a helpful parameter. It can be frequency or magnitude independent and allows comparing signals of different complexness. Several methods are available for the calculation of entropy. To determine the SE a frequency dependent approach is necessary. [Alve07]

ERMES ET AL. [ErPC08] define SE of an acceleration signal for a frequency band $[f_1, f_2]$ as

$$SE(f_1, f_2) = \frac{-\sum_{f_i=f_1}^{f_2} P(f_i)\log\big(P(f_i)\big)}{\log(N[f_1, f_2])}, \qquad \text{Equation 3-9 adapted from [ErPC08]}$$

where $P(f_i)$ represents the PSD value of the frequency f_i. The PSD values are normalized, resulting in a sum of one in the band $[f_1, f_2]$. The number of frequency components in the corresponding band in the PSD is $N[f_1, f_2]$.

In their work, ERMES ET AL. [ErPC08] used the SE for differentiation between running or walking and cycling. KHAN ET AL. [KLLK10] claim SE to be their best discriminating feature in recognizing resting, upper body, and lower body activity.

3.2.3. Miscellaneous features

Autoregressive coefficient

In signal processing, an autoregressive (AR) model is a representation form of a time series signal. It is an estimation and characterization of how the output variables of a signal depend on its own previous values. In addition, it has a direct link to the spectrum of the signal. [Dorf13]

Equation 3-10 defines the AR model of a random process $y(t)$ in the discrete time t, where $a_1, a_2, \dots, a_p$ are the coefficients of the model, p the order of the model, and $\varepsilon(t)$ the output uncorrelated error.

$$y(t) = \sum_{i=1}^{p} a(i)y(t-i) + \varepsilon(t) \qquad \text{Equation 3-10 [KhLK08]}$$

The number of past values $y(t-i)$, which were used to estimate the current value for $y(t)$, defines the order p of the AR model. [KhLK08]

The AR coefficient can be estimated with the Burg method. This method estimates the reflection coefficients and uses them to estimate the AR coefficients recursively. [Math13c]

DORFFNER evaluates AR coefficients as not exact descriptive, but sufficient estimation. The computation is rated as efficient. [Dorf13]

Signal magnitude area

The signal magnitude area (SMA) is defined as the area under the magnitude of the root mean square of all three axes [ChPS08]. It is calculated as

$$SMA = \sum_{i=1}^{N} (|x(i)|) + (|y(i)|) + (|z(i)|)\,, \qquad \text{Equation 3-11 [KLLK10]}$$

where $x(i)$, $y(i)$, and $z(i)$ represent the acceleration signal along x-, y-, and z-axis, respectively.

3.3. Classification methods

Classification is the allocating of elements to classes according to their characteristics. These characteristics are called features. In principle, elements with similar features belong to the same class. [Kram09]

Bayes classifier

A Bayes classifier is based on the Bayes theorem and used for probabilistic learning. It requires a probabilistic model or cost function, which estimates all misclassifications or unclassifiable data. Correct classifications do not occasion any costs. By minimizing these costs, a Bayes classifier has the lowest possible probability of error. However, to take observed data into account, independence assumptions have to be made which tend to be inaccurate. [DeSS11] [LoYA09]

Decision tree classifier

A decision tree consists of internal nodes, branches and leaves. Every internal node represents an attribute or feature. The branches are a test on the node, from which they are coming from. Every leaf represents an output class. Decision trees are constructed according to a training set. For this purpose, there are different algorithms available. A decision tree is applied to new data by running through the tree from the root node to a leaf. The ending leaf corresponds to the classification result. Decision trees describe relationships between features and output class by using simple decision rules. Therefore, they are easy to understand for the user. [Böhm03] [Cimi07]

A decision tree classifier is a decision tree used for classification. Numerous learning methods have been proposed. But most of them have a tree growing and pruning phase in common. [Dobr09]

Cluster analysis

Cluster analysis is an approach to find unknown classes (or clusters) in a set of data objects. Every object has a number of features. Clusters are found, where features of objects in the same cluster are as similar as possible and differentiate as much as possible from objects in other clusters. There are numerous clustering algorithms available for structuring a pool of features, which are based on unsupervised learning. [Prus06]

Hidden Markov model

A hidden Markov model is a generative probabilistic model for analyzing time-series data. At each time step, it consists of a hidden variable and an observable variable. This Markov process is based on two assumptions. First, a hidden variable depends only on the previous hidden variable. Second, an observable variable depends only on the hidden variable at that time step. Estimations of probabilities between observable and hidden variable allow a prediction and classification. [KNEK08]

Conditional random field

A conditional random field is a discriminative probabilistic model and has the same structure as the hidden Markov model. Instead of directed dependencies between variables, the conditional random field is an undirected graphical model. Conditional probabilities have been replaced by corresponding potentials. Estimating these potentials allows a classification. [KNEK08]

Artificial neural networks

The human brain has one of the most complex structures found in nature. Its ability to perform cognitive tasks is superior to modern computers, thanks to numerous neurons working parallel. The human brain consists of approximately 100 billion (10^{11}) neurons, which are connected with one another by 10^{14} to 10^{15} synapses. Stimulated neurons pass the signal on to other neurons. The signal of the stimulus can be amplified or reduced by the synapses (Hebbian theory). An artificial neural network (ANN) has a similar structure. It is a network of processing units with weighted connections to each other. [Kram09]

Learning in an ANN means to adjust the weightings of the connections. A distinction is made between supervised and unsupervised learning. During supervised learning, an ANN receives an example input and additionally information about the output result. Backpropagation is a supervised learning algorithm and will be explained later. Unsupervised learning is based on a self-organizing network without external feedback. The network is able to adjust its structure according to the input data. [Kram09]

Many types of ANNs are suited for classification problems. An ANN as a classifier requires a learning phase. It needs to be trained and learn how to classify chosen features. With supervised learning a training set of input data is available, containing example input features with known output class. In this phase, the classifier learns how to classify input data correctly with the aim to be able to apply this knowledge to new input data with unknown output. This ability to generalize knowledge requires a validation before practical use of the classifier. To this end, classification result and true class affiliation are compared and its error estimated. It is important that this testing input is not a subset of the training input. Too much training may result in overfitting. In this case, the classifier is able to classify the training data correctly, but new similar test sets tend to be misclassified. Its result is the unintended inability to generalize. [Kram09]

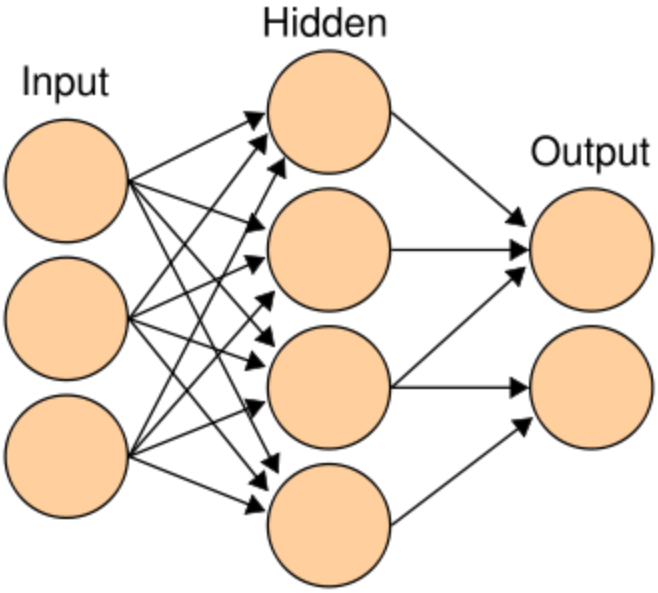


Figure 3-1: Artificial neural network topology

One of the most common learning algorithms is backpropagation. This method is based on supervised learning and requires a certain network topology. It usually consists of an input layer, an output layer, and hidden layers between them (see Figure 3-1). Units in the input layer receive the input features. The output layer units provide the response of the ANN to the input data. The connections between units are directed to the next higher layer and not recurrent from higher to lower layer. This kind of network is also known as feedforward network. The number of units per layer is dependent on the feature dimension and the output classes. This type of ANN is preferred for prediction and classification tasks. [Kram09] [AgBe00]

The basic idea of backpropagation is minimizing the percentage of misclassifications. Learning means reducing the error by manipulating the connection weightings. The name of this method refers to the way the error computed at the output layer is propagated backwards to the lower layers. In the first of two phases, the forward pass, initial values for the weightings are assumed. The input is propagated forward through the network to the output layer. Consequently, the classification result is generated and compared to the desired output. Its difference represents the error of learning. During the second phase, this error is propagated backwards through the network to the input layer, in order to change the connection weightings in such a way as to decrease the value of the error. These phases are applied repeatedly with a set of training data until the error converges or a time limit is exceeded. [Kram09] [BaHa00] [Kraw13]

3.4. Accelerometer

Accelerometers are sensors that measure acceleration. Acceleration is the rate of change in velocity in respect to time. It is a vector and has magnitude and direction. Accelerometers measure in units of g-force, which refers to the earth gravitational acceleration of 9,81 m/s². Accelerometers can measure vibrations, impacts, tilt angle, and motion of objects. [SENS08]

There are numerous types of accelerometers, which differ in principle of sensing and sensing element. Capacitive accelerometers sense a change in electrical capacitance, which changes between a static and dynamic condition of the sensor. A piezoelectric accelerometer is based on the piezoelectric effect. Certain crystals create an electric charge as external stress like acceleration is applied. Piezoresistive accelerometers measure the change in electrical resistance of a material under mechanical stress. Accelerometers based on the Hall Effect sense voltage variations in the magnetic field around the sensor. Magnetoresistive accelerometers have a similar structure and function, but instead of measuring voltage, they measure changes in resistance due to a magnetic field. Accelerometers based on micro-electromechanical systems (MEMS) technology are small structures with dimensions in micrometer scale. This technology is now being utilized to manufacture state of the art accelerometers. [SENS08]

GCDC accelerometer

As stated in Req10, the type of accelerometer to use is determined. *Gulf Coast Data Concepts* (*GCDC*) sell accelerometers based on MEMS. The 3-axis digital accelerometer sensors are manufactured by *Analog Devices*, Norwood, MA, USA. They are typi-

cally accurate to within 10 % using the conversion methods provided in the user manuals [GCDC10a] [GCDC13]. Most of these errors are due to an offset error, e.g. when the sensor in rest reports gravitational acceleration of 1,1 g instead of 1,0 g.

Figure 3-2: USB Accelerometer X6-2 (*GCDC*, Waveland, USA)

During software development, an USB accelerometer model *X6-1A* is used for the hip position, and an USB accelerometer model *X6-2* (see Figure 3-2) is used for the tibia position. Both have the same dimensions, but different types of power supply. The *X6-1A* is powered by a single “AA” sized alkaline battery, while the *X6-2* has an internal, hardwired 500 mAh lithium-polymer rechargeable battery [GCDC10b]. Since latter battery is rechargeable via USB cable, it is used for the tibia position, so the sensor can be attached permanently to the orthosis or shin guard. Replacing the battery would loosen the attachment of the sensor. Using a rechargeable battery ensures the sensors position and orientation. The *X6-1A* accelerometer is able to sample data at a minimal sampling rate of 10 Hz and the *X6-2* of 20 Hz. Both accelerometer models are capable of measuring accelerations between -6 g and +6 g. The data measured by the accelerometers is saved as raw count data. According to their user manuals [GCDC10a] [GCDC10b], the conversion algorithm is dependent on the settings of the accelerometer. The gain setting changes the proportionality coefficient between count data and g-force units (see Table 3-1). It is common to specify acceleration not in m/s² but in multiples of the gravitational force g (≈ 9.81 m/s²).

Table 3-1: Converting rules from raw counts data into g units [GCDC10a]

AD Resolution	Gain Setting	Deadband Counts
12-bit	Low	340×g
	High	1024×g

3.5. Evaluation methods

Confusion matrix

Proper function and correctness of a classifying system need to be validated and evaluated. A confusion matrix is a common method to evaluate performance parameters of a multiclass-classifier [Ting10]. In a two-dimensional matrix a true test data set is compared against its classification results. Test data with known classes or activities are used as input, because their results are required to fill out a confusion matrix. Table 3-2 presents an example of a confusion matrix for a three-activity classifier to differentiate between *Standing*, *Walking* and *Running*.

Table 3-2: Confusion matrix example

Confusion matrix		Classification result				
		Standing	Walking	Running	Accuracy	
True activity	Standing	10	0	0	100,0%	
	Walking	0	6	1	85,7%	
	Running	0	3	8	72,7%	Average Accuracy
	Reliability	100,0%	66,7%	88,9%		**86,15%**
				Average Reliability	**85,19%**	**85,71%** Overall Accuracy
Number of classifications		28				

The second row indicates that 7 data inputs were of the activity *Walking*. Six of those were classified correctly, one was misclassified as *Running*. The numbers of correct classifications are highlighted in green on the diagonal of the matrix. Off-diagonal elements contain misclassifications and are colored in red. The sum of a row is the total number of inputs of one activity. The sum of a column is the total number of classifications of one activity class.

For every row, the accuracy can be calculated as the number of correct classifications (green diagonal element) divided by the total number of inputs for this activity (sum of row). This leads to an accuracy of 85,7 % for the activity *Walking*. Accuracy is a measure of the degree to which the predicted activity of the classifier matches the true input data [SaWe11].

For every column, the reliability can be calculated as the number of correct classifications (green diagonal element) divided by the total number of classifications for this activity (sum of column). This leads to a reliability of 66,7 % for the activity *Walking*, meaning that two third of all *Walking* classifications actually represent *Walking* in the input data. Accuracy and reliability can be calculated for every activity and are written in yellow cells in Table 3-2. The averages of these values are performance parameters of the classifier.

Average accuracy and average reliability are highlighted in grey cells. A third parameter is the overall accuracy. It is calculated as the total number of correct classified activities (diagonal elements) divided by the total number of input activities (number of classifications) [ITCI13].

Usability questionnaire

A new GUI has to be developed for the custom software. To evaluate the usability of the software and its user interface, a full usability test would be appropriate. However, such a test in all its dimensions is too time consuming and beyond the scope of this thesis. Therefore, a rough quantification of the ease of use is accomplished by a short usability questionnaire.

The conducting researcher of the *NutriHEP* study is asked to read the user manual (see appendix 9.2) and to work on the software. If the user manual is not helpful for unpredictable problems, the developer will interfere. After completing several tasks, the researcher is asked to fill out a usability questionnaire (see appendix 9.1). This questionnaire was designed to give a short overview of the impressions of the software to the user.

The results can be sighted in appendix 9.1 and are not described separately. However, the discussion of the results will be covered in chapter 7.1.

3.6. State of the art

This chapter describes the state of the art in human activity pattern recognition from acceleration data.

An approach of activity recognition in a home environment with inexpensive annotation tools and easy installation was developed by VAN KASTEREN ET AL. [KNEK08]. In their work, activities like showering, breakfast, dinner, or sleeping were under consideration. For this, digital state change sensors were installed in a subject's home. A Bluetooth headset was worn by the subject to annotate the current activity, thus, generating a ground truth data set with known output. Raw, point-of-change, and last-observation data was classified with hidden Markov models and conditional random fields. VAN KASTEREN ET AL. achieved the best reliability of 95,6 % and the highest accuracy of 79,4 % with conditional random fields and hidden Markov models, respectively.

CHUNG ET AL. [ChPS08] focused in their publication on a real-time application of activity recognition. With wireless MEMS accelerometers attached to a subject's chest, three kinds of activities (resting, walking, and running) were classified. SMA and median frequency were the only two features used as input for a decision tree. The ground truth data for evaluation purposes was recorded by a following annotating researcher. An accuracy of 81,25 % was achieved.

The focus of LONG ET AL. [LoYA09] was the comparison of a Bayes classifier and a decision tree classifier in assessing daily energy expenditure. In this work, activities like walking, running, cycling, and driving were classified. In a naturalistic environment without researchers, acceleration data was acquired with a single tri-axial accelerometer placed on the waist. Annotations of activities for a ground truth data set were written down by subjects. The allocating of acceleration data to activities was done by hand. SD, periodicity, and orientation features were input for a decision tree classifier and Bayes classifier. Accuracies of 72,8 % and 71,5 % respectively were achieved.

KHAN ET AL. [KLLK10] were able to identify new features, which allow activity recognition independent of the sensors position. The intention was to establish a recognition system for an unsupervised free-living environment. A single tri-axial accelerometer was worn either in a chest pocket, in a trousers pocket (front or rear), or in an inner jacket pocket. Classified activities include resting, walking, climbing stairs, running, cycling, and vacuuming. Again, a ground truth data set was ensured by having the subjects use a Bluetooth headset with speech recognition. SE, AR coefficients, and

SMA were the features of choice. A two-level recognition approach included ANNs and linear discriminant analysis. An accuracy of 90 % was achieved.

LEE ET AL. [LKK+10] established a real-time activity recognition system, which would serve as personal log life system. Every day activities were distinguished, including driving and climbing stairs. Rather impractical non-wireless accelerometers were attached to the chest. To generate a ground truth data set, camera recordings were analyzed. Classification features were SD, SE, correlation among all axes, AR coefficients, SMA and tilt angle. Furthermore, stride length and step count were extracted for calculating distance, speed, and energy expenditure. LEE ET AL. constructed a two-level classifier, which first distinguishes between static and dynamic activities. On the second level, two ANNs (static and dynamic) were trained with backpropagation algorithm. An accuracy of 84,8 % was achieved.

In the *NASA Glenn Research Center*, HANSON ET AL. [HGP+11] worked on a new method for tracking crewmember activity during space missions in reduced gravity environment. They employed an enhanced zero gravity locomotion simulator to measure different types of locomotion under different gravitational influences. Two wireless accelerometers, attached to footwear and waist, measured activities like walking and running under earth gravity. Furthermore, walking, running, hopping, and loping under 1/6 earth body weight simulated activities under moon conditions. Ground truth data was established by demanding certain activities. Features like signal peak, spread, frequencies, power, and centroid were used as input for an ANN. It achieved an accuracy of 100 %.

KWAPISZ ET AL. [KwWM11] developed an activity recognition system with phone-based accelerometers. Via a custom built smart phone application, data was acquired from the built-in accelerometer. It was able to recognize walking, jogging, climbing stairs, sitting, and standing. For generating a ground truth data set, the subjects were asked to label every activity using the installed application. Mean, SD, root mean square, time between peaks, average absolute difference, and binned distribution were the features of choice. KWAPISZ ET AL. compared decision trees, logistic regression and ANNs as classifying method. These methods reached accuracies of 85,1 %, 78,1 %, and 91,7 % respectively.

4. Design concept

Different body movements cause different accelerations on several body parts. Every movement appears to have its individual characteristic pattern in acceleration. An identification of acceleration patterns and their proper allocation to human activities can be approached with several classification methods (see chapter 3.3). This chapter explains the choice of the classification method and the data flow process of the activity recognition.

4.1. Classification with artificial neural networks

ANNs are the method of choice for human activity pattern recognition. In the following, it will be explained why ANNs are more suitable in this thesis than the other classification methods described in chapter 3.3.

A Bayes classifier is based on naïve independence assumptions. Those can be wrong, but this does not imply that the classifier will fail. Instead, the classification performance will be poor, because the features tend to have a similar distribution, and differences are only in relations of the features. A Bayes classifier is not suitable, because there is no knowledge about the feature distribution and the relations. Their negative effects on the classification performance cannot be ruled out. [Böhm03]

LONG ET AL. found out that, in general, a decision tree has the best performance in activity classification. However, from a practical point of view, this classification is not very suitable, because it needs a lot of implementation. If new activities need to be incorporated, a completely new decision tree classifier needs to be re-built. Manual tuning by experts might also be necessary, because tree training is completed on isolated activity events. This results in a low level of extensibility. [LoYA09]

A main disadvantage of cluster analysis is the readability of the results. They are not easily comprehensible and need some kind of interpretation [HeKi08]. Furthermore, it is an unsupervised learning algorithm, which, according to WAGSTAFF, uses *"very general notions to identify patterns and interesting structure in data"* [Wags02]. Unsupervised learning algorithms are unguided, and may tend to focus on uninteresting patterns (e.g. patterns due to systemic errors). A supervised learning algorithm is more suitable to the problem of human activity pattern recognition, because known activities are to classify.

ANNs grew from basic research to establishing practical implementations of innovative applications [Karr10]. Where linear models are unable to describe a model accurately and the underlying relationships are not well known, a nonlinear neural net-

work is helpful [HrLe05] [Tu96]. ANNs have a proven record of success in human activity recognition [HGP+11] [KLLK10] [KwWM11] [LKK+10]. Additionally, they are easy to conceptualize, because numerous libraries and implementations are available. However, an ANN works as black box, because connection weightings of a trained ANN do not reveal relationships between input features. But if the classification result is in focus, this issue is considered as reasonable compromise.

ANNs were often blamed for randomness in results and being unpredictable. The latter accusation is without any reason, because same inputs always give the same results as long as the weightings do not change. Consequently, the randomness concerns the training process of the ANN. ANNs are initialized with random weightings, i.e. the same training set can lead to different ANNs. However, if the validation error during training converges and barely changes, the effect of randomness is marginal and has no effect on the performances of an ANN. ANNs have a very robust performance. However, the process of training is hard to comprehend. [DeSS11]

4.2. Data flow process

The classification process with ANNs can be divided into two phases (see Figure 4-1). In the first phase of training and testing, acceleration data is analyzed and its features are extracted. The data with known output is randomly separated into training set and testing set. The weightings of the ANN are adjusted until the validation error converges. The result is a trained ANN, which is capable of classification of activities. The second phase begins with the acquisition of new acceleration data with unknown classes. Its features are extracted with the same functions as before. The trained ANN is then fed with these input features and classified activities are the result.

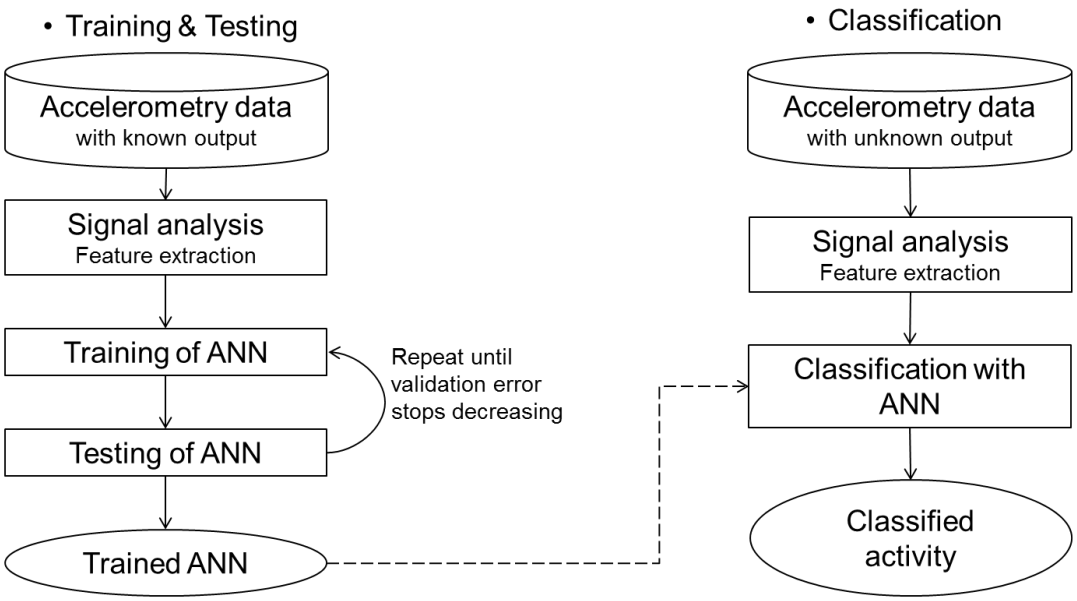


Figure 4-1: Training and classification concept

For each subject, acceleration data is acquired from two sensors, placed parallel to the tibia and close to the hip. Both data sets run through the same activity classification algorithm, which is shown in Figure 4-2 as a flow diagram. The consecutive processes of data acquisition, dynamic nature classification, feature extraction, and activity classification are visualized.

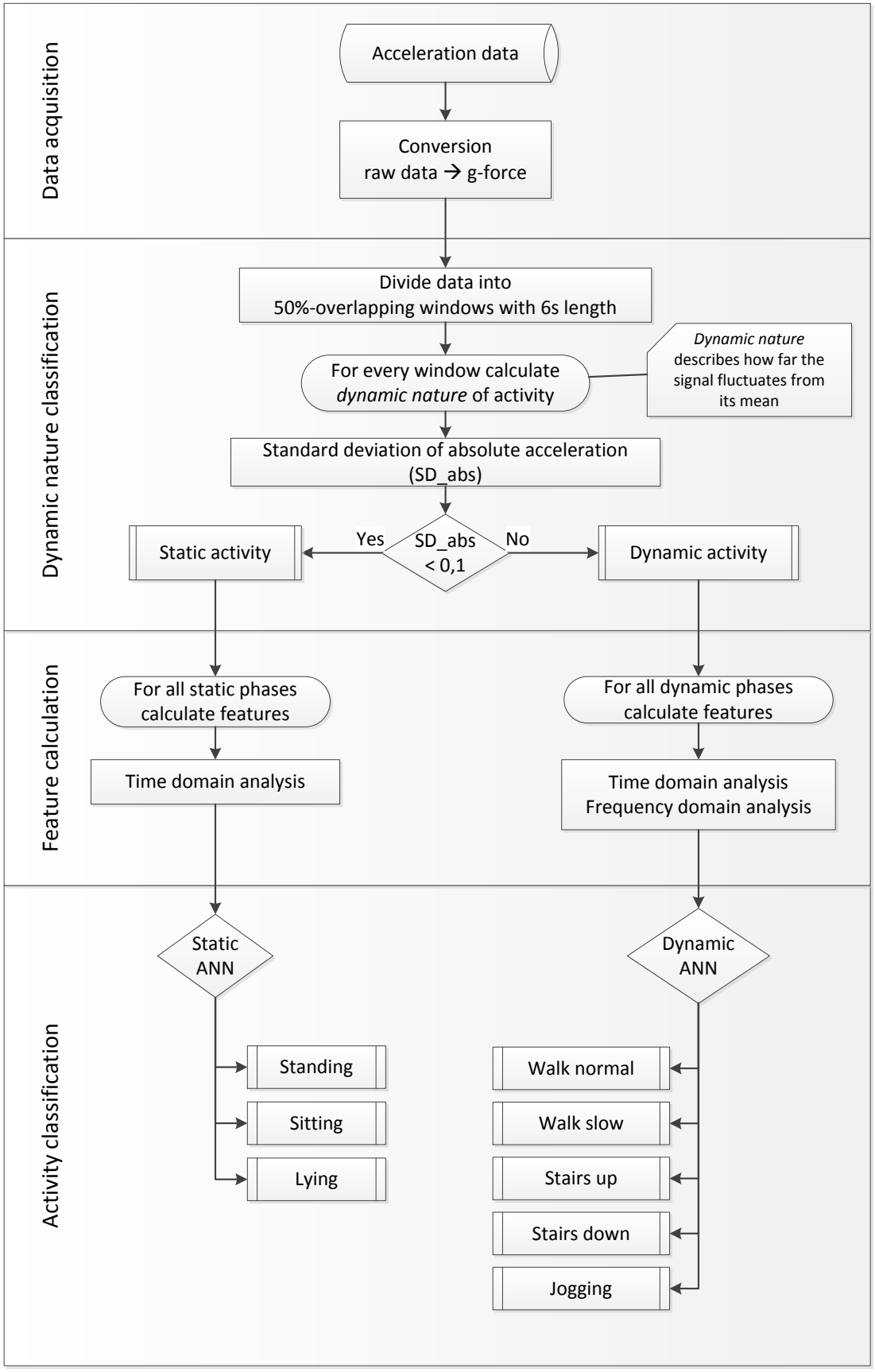


Figure 4-2: Data flow diagram

4.2.1. Data acquisition

Accelerometer

MEMS accelerometers are used from *GCDC* (see chapter 3.4). Their factory calibration with 10 % accuracy is sufficient, because acceleration values, or more precisely extracted features, are compared relatively against each other. An offset error, which affects the entire measurement, does not have an effect on the classification results.

GCDC write in their user manual that a lower sampling rate extends the operating time of the sensor [GCDC10b]. This suggests keeping the sampling rate as low as possible. BOUTEN ET AL. [BKV+97] found frequencies up to 20 Hz to be sufficient to assess daily physical activity using body-fixed accelerometers.

Further studies reviewed in [BKV+97] recommend an ability to register accelerations within an amplitude range of -12 g to +12 g for ankle placed sensors and a level of -6 g to +6 g at waist. Both accelerometer models from *GCDC* are capable only of the latter range. This would indicate a problem concerning the sensor at the tibia position. However, these specifications are based on data acquired by BHATTACHARYA ET AL. during tread mill runs with speeds up to 11 km/h [BMSG80]. In this thesis, light jogging is the only type of running to be measured. Looking at the amplitude range of several subjects lightly jogging, the accelerations at the tibia position did not exceed the range of -6 g to +6 g.

In conclusion, the accelerometers are set to a sample rate of 20 Hz and a low gain, so the detecting ranges from -6 g to +6 g. These settings allow assessing physical activity and additionally prolonging battery life.

Conversion

The data measured by the accelerometers is saved as raw count data. Its conversion into acceleration data is inevitable. Gain settings change the proportionality coefficient between count data and g-force units (see chapter 3.4). To make the software program applicable for unknown future use, the conversion process is designed to choose the corresponding proportionality coefficient according to the received data.

Signal noise can affect the outcome of the analysis. Such interference can be lowered by filtering the data before the analysis. Literature review has shown that several studies have applied different filter methods during pre-processing acceleration data. HANSON ET AL. [HGP+11] used two accelerometers simultaneously, sampled the data with 256 Hz (waist sensor) and 1024 Hz (foot sensor), and processed each data set with a low-pass filter at 100 Hz. A sample rate of 2000 Hz and filtering by a fourth-order low-pass filter with a cut-off frequency of 45 Hz was chosen by GENC ET

AL. [GeMC06]. Due to these chosen sampling rates, high frequency noises are present and need to be filtered out. When a lower sampling rate is chosen, no antialiasing filter is necessary on the accelerometry data [PGKH09]. A moving average filter of order three was incorporated by KHAN ET AL. [KLLK10] to filter out random noise. But with a sampling rate of 20 Hz an average filter would smooth signal peaks, which may contain important information about the acceleration within an activity. Therefore, no moving average filter is applied.

In conclusion, the pre-processing contains the conversion of the accelerometer's raw count data into g-force data according to the *GCDC* user manual [GCDC10a]. No kind of filtering is required, due to a low sampling rate of 20 Hz, which is sufficient to asses daily physical activity [BKV+97].

4.2.2. Dynamic nature classification

Windowing

The signal is divided by 50 %-overlapping moving windows with a size of 6 seconds. In this work, human activity, like walking or climbing stairs, is considered as physically effective within a minimal lasting time of 6 seconds. Furthermore, a larger window size results in a less precise time allocation of activities. However, a small window size is unrewarding, because low numbers of samples affect the frequency analysis. An overlap of 50 % was proven to be successful by RAVI ET AL. [RDML05] and LEE ET AL. [LKK+10]. Additionally, unpublished tests with the window size were established and came to the same result during the work with WEBER ET AL. [WDY+13]. The following calculations are applied to every window data.

Classification of dynamic nature of activity

Dynamic nature describes the dispersion of an acceleration signal. The dynamic nature of a human activity can be either static or dynamic. Static activities are activities, where no movements of the interested body parts are noticeable (e.g. sitting or standing). Dynamic activities are activities, where cyclic movements can be assumed (e.g. walking or climbing stairs). This differentiation is needed, because feature extraction methods in the frequency domain are not applicable to acceleration signals of static nature. The SD of the absolute acceleration is considered as an easy-to-calculate parameter. The absolute acceleration is the effective resulting value of all three spatial axes and is approached by using the Euclidian norm calculation (see chapter 3.2.1). SD was proven to be a good quantifier of an activity's dynamic nature by FRANK ET AL. [FNRA10] and WEBER ET AL. [WDY+13], because the SD correlates with the dispersion of a signal.

A threshold is needed to distinguish the two types of dynamic nature. During the work of WEBER ET AL., task related mean SDs for several activities were identified. Unpublished observations have shown that an SD threshold of 0,1 g is appropriate to differentiate between static (<0,1 g) and dynamic (>0,1 g) activities. This decision rule is applied for the classification of the dynamic nature of human activity. [WDY+13]

4.2.3. Feature calculation

Static activity classification

For the entire time phase where the activity is of static nature, time domain features are extracted as described in chapter 3.2.1. Frequency domain features are not applicable in this case. Descriptive methods, like mean, SD, and absolute maximum (see Table 4-1) are considered as *"traditional features that are used for acceleration activity recognition"* [GjGC10] and have been successfully applied by HANSON ET AL. [HGP+11], PREECE ET AL. [PGKH09], BOUTEN ET AL. [BKV+97] and many more.

Dynamic activity classification

For the classification of dynamic activities, time and frequency domain features are extracted.

The peaks of the power magnitude represent the main frequencies and are quantified by the first components of the FFT analysis. It seems reasonable to investigate the frequencies in activity patterns, because different types of locomotion underlie different frequency patterns. PREECE ET AL. [PGKH09] showed in their work that the magnitude of the first five components of an FFT analysis have the best classification accuracies, with sensors worn on waist and ankle. However, in this thesis, the first three components are used as describing feature. Furthermore, the periodicity was a promising feature for Hanson et al. [HGP+11] and Lee et al. [LoYA09].

SE is a feature, which was mainly introduced by KHAN ET AL. [KLLK10] [LKK+10]. They claimed it to be their best discriminating feature in recognizing resting, upper body, and lower body activity. ERMES ET AL. [ErPC08] used SE for differentiation between running or walking and cycling.

KHAN ET AL. [KLLK10] also used the AR coefficients and the SMA as describing features and produced promising results. In a preceding work, KHAN ET AL. [KhLK08] found with autocorrelation analysis that an AR model of order 3 is most suitable. Additionally, CHANG ET AL. [ChPS08] found different activities, like running and walking, have different SMA levels. Whereas KHAN ET AL. [KLLK10] significantly improved the recognition rate of dynamic activities with a combination of SMA with AR coefficients. It

was the best discriminating feature for all activity classes and all sensor positions. A proportional correlation between SMA and energy expenditure was approved by MATHIE ET AL. [MCLC04] and BOUTEN ET AL. [BKV+97].

Table 4-1: Feature list

Domain	Feature	Classification of
Time domain	Mean	Static and dynamic activities
	Standard deviation	
	Absolute maximum	
Frequency domain	First 3 peaks in power magnitude	Dynamic activities
	Spectral entropy	
	Autoregressive coefficient	
	Signal magnitude area	

4.2.4. Activity classification

Classification with neural network

The hierarchical order of multiple ANNs is based on the structure approached by LEE ET AL. [LKK+10]. Their system consisted of three ANNs: state recognition, static activity recognition, and dynamic activity recognition. In this thesis, a similar structure is used. Instead of the state recognition ANN, a SD decision rule is implemented. This lowers the complexity of the classifier and the training process.

The settings of the ANNs basically go according to HANSON ET AL. [HGP+11]. The hidden layer has a size of 20 neurons. HANSON ET AL. stated that a difference of ± 5 neurons did not affect the results. The training data set is divided randomly into training (70 %), validation (15 %) and testing (15 %) set. Scaled conjugate gradient backpropagation is used as training function and the mean square error is the performance parameter, which has to converge.

Compliance check

A decision rule is incorporated to estimate the subject's compliance. The check for compliance is only applicable during the *NutriHEP* study. In collaboration with the conducting researcher, scenarios were elaborated concerning the wearing of the *HEP* orthosis. With the activity classification described above, a differentiation between static and dynamic activities is possible. Looking at one time window, there are four possible combinations of classifications:

1. Tibia and hip sensor both measure dynamic activities
2. Tibia and hip sensor both measure static activities
3. Tibia sensor measures dynamic activity and hip sensor measures static activity
4. Tibia sensor measures static activity and hip sensor measures dynamic activity

In the first two cases, the classifications seem consistent and the subject is either moving or not in motion at all. In the third scenario it can be assumed that the hip sensor is not attached. Other explanations could be a sitting subject moving the leg or doing indoor cycling. Nevertheless, it appears plausible that in the third scenario the subject is still wearing the orthosis. The fourth scenario can be interpreted as a moving upper body and static lower body. This combination of body movement is difficult to realize. Attempts in generating such scenarios (e.g. with hula hoops) failed, because the tibia sensor always detected dynamic activities as well. This finding leaves the explanation of an unattached tibia sensor. In fact, the tibia sensor is permanently connected to the orthosis, so the fourth scenario indicates that the orthosis is not worn.

The decision rule is not applied to every single pair of tibia and hip classification. A window of 240 samples (= 12 seconds) is checked, if the fourth scenario applies. If it applies for all pairs of samples, the orthosis is considered as not worn.

5. Implementation

This chapter describes the programing structures, which were developed to implement the classification concept. All tools and programs used during this thesis are listed. The main features of the software and their algorithms and basic functions are explained in more detail.

5.1. Implementation environment

The software development uses the following external software:

- *MATLAB* R2012b v8.0.0.783 (*The MathWorks, Inc.*, Natick, MA, USA)
- *Neural Network Toolbox* (*The MathWorks, Inc.*, Natick, MA, USA)
- *Signal Processing Toolbox* (*The MathWorks, Inc.*, Natick, MA, USA)
- *MATLAB Compiler* (*The MathWorks, Inc.*, Natick, MA, USA)
- USB accelerometers *X6-1A* and *X6-2* (*Gulf Coast Data Concepts*, Waveland, MS, USA)

5.2. Preparation phase

This chapter explains the implemented code in *MATLAB* for the software functions which are necessary beforehand. Only fundamental code lines are presented here. The full program code is accessible on the appended CD.

5.2.1. Data import

Function name: retrieveData.m

This function allows automatic storing of acceleration data from a USB drive to a hard drive. The GUI language is chosen by the calling main function. The user selects the USB root drive. Its details and header information are retrieved. The function automatically sorts the CSV-files in the following folder hierarchy: subject ID, calendar week, and sensor position. This structure makes it easier to compare activities between different weeks. The files are renamed with date and time of starting acquisition. This avoids overwriting of different data sets with the same naming.

Read out USB Sensor details from configuration file and header

The function checks for a TXT-file named “config.txt”. It contains information about the subject and the body position the accelerometer was worn. How this information is

changed in this file is explained in the user manual (see appendix 9.2). If no configuration file is found, the software throws an error, saying that a wrong drive was chosen. Starting time and date of the recorded data is read out through the data files themselves. Every data file contains a header with such information. This function is exclusively adapted to *GCDC* accelerometers.

Import files from USB Drive

After checking for a valid saving path, the number of files for import is gathered. A generated progress bar, showing the computing progress according to the number of imported files, is shown on the screen. For every file, date and time of acquisition are read from the header. According to the date, the calendar week number is calculated with an external function[1]. If the corresponding folder is not existent, it will be generated. After that, the data is renamed and copied to the right target folder. Any error in the process described above, i.e. missing header information, will be noted in a list of failed imported data. This list will be shown, after trying to copy every data file from the chosen accelerometer. An abortion of the saving process is possible by closing the loading progress bar window.

5.2.2. Time synchronization

Function name: timeSynchro.m

This function synchronizes two accelerometers, which experienced one shared event (e.g. a light hit against each other). The separating time shift of both sensors is estimated by the time difference of the peaks caused by the shared event. First, the function asks for two CSV-files, which recorded the shared event. The input of the sensors' IDs is mandatory. Then, the user is asked to mark the peaks of the shared event. The function calculates the time shift, saves it in a DAT-file, and plots the time synchronized acceleration data for visual validation.

Import of two data sets of two different sensors

Both data sets are imported with the `RAW_Conversion.m` function (see appended CD). The naming of the sensors should be the hardware serial number or a unique institutional naming, assuring an unambiguous identification. Consequently, naming the sensors with the same ID is not possible.

[1] http://www.mathworks.com/matlabcentral/fileexchange/22663-weeknum/content/weeknum.m

Calculation of absolute acceleration on all axes

To identify the peak of the shared event in the acceleration data, the absolute acceleration on all axes is plotted. For its calculation the Euclidian norm is used (see chapter 3.2.1).

```
%calculate the absolute acceleration
m_A = length(dataX_A);
dataAbs_A = zeros(m_A,1);
for i = 1:m_A
     dataAbs_A(i,1) = sqrt(dataX_A(i)^2 + dataY_A(i)^2 + dataZ_A(i)^2);
end
```

Choose and save timestamp of peak

After plotting the absolute acceleration on the screen, the user has to click on the peak of the shared event for identification. This has to be done for both sensors in a specific order. The software will indicate that the first signal is always the blue line. The times of the chosen data points are saved for the following step.

Calculate time shift

The time shift is equal to the time difference between the saved time stamps in the step before. The difference is calculated by the built-in `etime` function, which estimates the elapsed seconds between two date vectors. The time difference is multiplied by a thousand and rounded to an integer number to provide a time shift in milliseconds. This conversion is needed for proper function of the following step.

```
delta_t = floor(etime(datevec(TimeA),datevec(TimeB))*1000);
```

Apply time shift on time array

Once the time shift is known, the time array can be manipulated. The shifting milliseconds are added to every date vector with the built-in function `addtodate`. The smallest time dimension is millisecond and the added time has to be an integer number.

Visual validation and saving results

For a visual validation by the user, both time signals are plotted again with one being shifted. If the results are unsatisfying, i.e. the peaks of the shared event are not matching in time, the time synchronizing process has to start over. If the results are satisfying, the calculated time shift is saved in a DAT-file for future processing. It will be used during the activity classification in chapter 5.4.1.

5.3. Training phase

This chapter explains the implemented code in *MATLAB* for the software functions which are necessary during the training phase of the ANNs. Only fundamental code lines are presented here. The full program code is accessible on the appended CD.

5.3.1. Data conversion

Function name: multiple_RAW_Conversion.m

This function executes an import of all CSV-files in one folder. The raw data is converted into g-forces. If more than one file is present, the alphabetical order must correspond to the chronological order of the files. First the number of input arguments is checked, since this function has default inputs. If no pathname is given, a folder has to be selected. The first file is imported and its header information is retrieved. The conversion from raw count data into g-force data follows the instructions in the manual from *Gulf Coast Data Concepts*. Finally, the absolute acceleration is calculated, and if needed, everything gets plotted for visual representation.

Input validation

Two input variables are configurable, plot request and pathname. If a plot is requested, this variable has the value `1` (default value). Any other value will not result in a plotting screen at the end of this process. The pathname defines the folder containing the data files for conversion. If no pathname is given, it has to be selected via a pop-up menu.

Import accelerometer raw data

All CSV-files are imported at once with the built-in `dir` function. From the first one, its starting time, sampling rate and gain are read out from the header information. These details are important for the conversion process and higher level functions.

Convert raw data to g-forces

According to the gain setting the correct calibration is chosen from the user manual (see Table 3-1 on page 18). For a high gain setting, which is used to acquire accelerations between -2 g and +2 g, the raw count data is divided by 1024. For a low gain setting, allowing acquisition in a range from -6 g to +6 g, the raw count data is divided by 340. These calculation specifications are taken from the user manuals of *X6-1A* and *X6-2* accelerometers by *GCDC* [GCDC10a]. The time stamps, saved in the CSV-files, are also retrieved and saved as time array. In the accelerometers time is saved as elapsed seconds since the start of the measurement. Therefore, the relative time array

needs to be converted into an array with absolute serial date numbers. This is accomplished by adding the start time to every time stamp with the built-in `addtodate` function. Again, the seconds need to be converted into milliseconds.

```
% Read out time array
rawT = raw.data(:,1);
% convert relative time array to absolute serial date number array
dataTime = zeros(length(rawT),1);
for i = 1:length(rawT)
     dataTime(i) = addtodate(startTime, floor(rawT(i)*1000), 'millisecond');
end
```

Import and convert rest of accelerometer raw data

The remaining CSV-files are imported and converted in the same manner as before. Start time information is retrieved from every single header. An abortion of the conversion process is possible by closing the loading progress bar window.

Plotting graphs of acceleration

If a graphical visualization is requested, absolute acceleration in one window and all axes individually in the other window are plotted. To this end, the purpose-built functions `plotAbs.m` and `plotXYZ.m` were programmed (see appended CD).

5.3.2. Dynamic nature classification

Function name: dyn_natureClassification.m

This function classifies the dynamic nature of an acceleration signal, whether it is of static or dynamic nature. To this end, the SD of the absolute acceleration is calculated. If a certain threshold is exceeded the activity is considered as dynamic.

Classification using standard deviation

The threshold for differentiating between dynamic and static activities is set to 0,1 g. For every moving window the SD of the absolute acceleration is calculated with the built-in `std` function. With an if-else statement the dynamic nature is saved as static (`0`) or dynamic (`1`). Additionally, the time stamp in or next to the middle of the data window is saved.

```
threshold = 0.1;
% calc SD of absolute acceleration
if std(winData) < threshold
     dyn_nature(2,i) = 0;            % static activity
else
     dyn_nature(2,i) = 1;            % dynamic activity
end
% take timestamp in/next to the middle of window
dyn_nature(1,i) = dataTime(winStart+halfWin-1);
```

5.3.3. Feature extraction: time domain analysis

Function name: calcTime_movWin.m

This function extracts features of the time domain for classification purposes. First, the data is divided into 50 %-overlapping moving windows. For every window mean, SD, and absolute maximum of the data is calculated.

Calculate features for every window

All features in the time domain can be calculated with the built-in functions `mean`, `std` and `max`. These three features are assembled in a feature matrix where every row contains the features of one moving window. This is applied to every axis of the acceleration data.

```
for i = 1:N
% calc mean of one axis
     meanValue = mean(winData);
% calc standard deviation of one axis
     stdValue = std(winData);
% calcabsolut maximum acceleration in one axis
     maxValue = max(abs(winData));
% assemble features as array
     featMatrix(:,i) = [meanValue; stdValue; maxValue];
end
```

5.3.4. Feature extraction: frequency domain analysis

Function name: calcFFT_movWin.m

This function applies an FFT on acceleration data and extracts frequency domain features for classification purposes. First, the data is divided into 50 %-overlapping moving windows. For every window an FFT is performed. The power of the FFT is calculated and the three highest peaks are considered. Their magnitude and frequency are saved in a feature matrix.

Calculate features for every moving window

The transformation into the frequency domain is approached with the built-in `fft` function. The first component of the result is the sum of the data and can be removed [Math13a]. The length of time in seconds of the window is calculated. As described in chapter 3.2.2, the power of a signal is the squared Fourier transform. The scaling frequency goes up to the length of time in Hertz. With the help of the built-in function

`findpeaks`, the first three peaks in power and the corresponding frequencies are determined. They are saved in a feature vector in descending order of the power.

```
% calc FFT
    f = fft(winData);
    f(1) = [];
    n = length(f);

% calc time length of acquisition
    T = (n+1)/rate;

% calc power of FFT analysis
power = abs(f)/(n/2);
power = power(1:floor(n/2)).^2;
freq = (1:n/2)/T;

% find peaks
[pks,locs] = findpeaks(power,'sortstr','descend');

% save first three peaks sorted in feature array
Npeaks = 3;
feat_vect = [pks(1:Npeaks); freq(locs(1:Npeaks))'];
feat_mat = [feat_matfeat_vect];
```

Function name: calcKhan_movWin.m

This function calculates the following features from accelerometry data: SE, AR coefficients, and SMA. All calculations are done according to KHAN ET AL. [KLLK10]. Therefore, this function is called `calcKhan_movWin.m` and the features are referred to as Khan Features in the software code. First, the data is divided into 50 %-overlapping moving windows. Furthermore, the SE on all three axes is calculated by a sub-function `calc_SE.m`. The AR coefficients are estimated by a built-in *MATLAB* function using Burg's method. For the SMA the sums of the absolute accelerations on each axis are summed up (see chapter 3.2.3).

Calculate Khan features for every window

The SE is calculated for every axis by the sub-function `calc_SE.m`. This function calculates the SE of the acceleration signal for a certain frequency band. The calculation is done according to KHAN ET AL. [KLLK10] as described in chapter 3.2.2. After an FFT the PSD is calculated. SE is calculated for each window as in Equation 3-9 in chapter 3.2.2. The FFT is calculated as described before. For the estimation of the PSD the built-in function `dspdata.psd` is used. This function generates the PSD values and

the sum at all frequencies can be calculated. The division of this sum by the logarithm of the number of frequency components in the corresponding band gives the SE.

```
% spectral entropy (SE)
SEX = calc_SE(winX,rate);
SEY = calc_SE(winY,rate);
SEZ = calc_SE(winZ,rate);
```

The AR model is estimated with Burg's method and the coefficients are computed by the built-in `arburg` function. According to KHAN ET AL. [KhLK08] the most suitable model is of order 3.

```
% autoregressive coefficient (AR)
% estimating AR model coefficient based on Burg's method
% with order = 3
ARX = arburg(winX,3);
ARY = arburg(winY,3);
ARZ = arburg(winZ,3);
```

The SMA is calculated according to Equation 3-11 in chapter 3.2.3. Basic arithmetic operations are used as shown below.

```
% signal magnitude area (SMA)
SMA = sum(abs(winX)) + sum(abs(winY)) + sum(abs(winZ));
```

5.3.5. Neural network training

Function name: main_train.m

This chapter explains the main function for the training of an ANN from acceleration data. It is adjusted for the use during the *NutriHEP* study, thus, combining the training of two ANNs (static and dynamic) for two accelerometers (hip and tibia). According to Req9, a training function for a single accelerometer is available. It is based on the same algorithm principles and the code is adjusted to one ANN for one sensor position. The main function is called `main_train_single.m`.

A standardized activity catalog and its number of static activities are declared as global variables. The user chooses a folder containing the training data from hip and tibia position. It is assumed that the second upper folder name corresponds to the subject's ID, which is ensured by using the import function (see chapter 5.2.1). The ANNs for the tibia and for the hip are trained and saved.

Initiation of global variables

At first, two variables, `activityClass` and `Nstatic`, are declared as global variables. Their validity for all sub-functions is essential for adding new activities to the catalog. The activity catalog contains the names of all activities which are to be classified. It is saved as a list in `activityClass`, where the first elements are of static nature. If `Nstatic` is 3, then the first three elements in the activity catalog are static activities. The implementation of a variable activity catalog is essential for the unknown use with a single accelerometer.

```
global activityClass
global Nstatic

% Set standardized activity catalog
activityClass = {'standing','sitting', 'lying', ...
'walking normal', 'walking slow', ...
'stairs up', 'stairs down', ...
'jogging', 'sprinting'};
Nstatic = 3;
```

Select data

The user selects a folder with training data set for hip and tibia position via a browsing window. The subject ID is read out from the folder path. A new function `train_dynNN.m` is called to start the training for tibia and hip data individually. The trained ANNs are saved in the training folder.

Function name: train_dynNN.m

This function trains an ANN with training data, which is already allocated to activities. Acceleration data is imported and its dynamic nature is classified. For the training of the ANN, the feature extraction uses a window size of 6 seconds. For the actual training and testing of the ANN, a sub function generated with *MATLAB* is used (`trainNN.m`).

Classification of dynamic nature of acceleration data

After importing the raw acceleration data from a selected folder, the dynamic nature is classified by the function `dyn_natureClassification.m` (see chapter 5.3.2).

Training of the neural network

For the supervised learning algorithm, a training set with known output is necessary. The allocation of activities to acceleration data is done by the user via the GUI. To start this feature, the function `selectTrainData_template.m` is called. The allocated

training set is saved in the training folder, and its features can be extracted. This is computed by the functions described in chapter 5.3.3 and 5.3.4. Additionally, the allocated activity is saved in the feature vector. One feature vector contains the features for a 6 second window and the index number of the activity in the activity catalog. All the features from one training set are concatenated in one matrix. It is called target matrix, because it contains the target output of the classification. These target matrices serve as input for the function `trainNN.m`.

Function name: selectTrainData_template.m

This function saves acceleration data with its actual activity information for training purposes. The absolute acceleration is plotted for visual representation of the signal. A hard coded template is drawn, consisting of several activities with specific lengths of time. This template corresponds to the activity course, which will be explained in chapter 6.2.3. The user can adjust these templates to the acceleration signal. With one click all these templates are saved. If necessary, additional activities can be added and identified for the training of an ANN.

Plot graph with absolute acceleration and dynamic nature classification

After importing the acceleration data, the absolute acceleration from all three axes is plotted as a blue graph. Within this graph, the dynamic nature is also visualized. Red dots distinguish between static (0) and dynamic (1) activities (see Figure 5-1). The pink and green vertical lines represent the activity template. They can be dragged by the user to allocate the flagged activity. Flags are at the top of every second vertical line, indicating the beginning of an activity. Pink lines are dynamic and green lines are static activities.

When the template is saved, a new folder is created named "`TrainingData`". It contains sub-folders called "`static`" and "`dynamic`". In each folder the associated activities are saved. Acceleration data of one activity is saved together in one file, which is given the activity name.

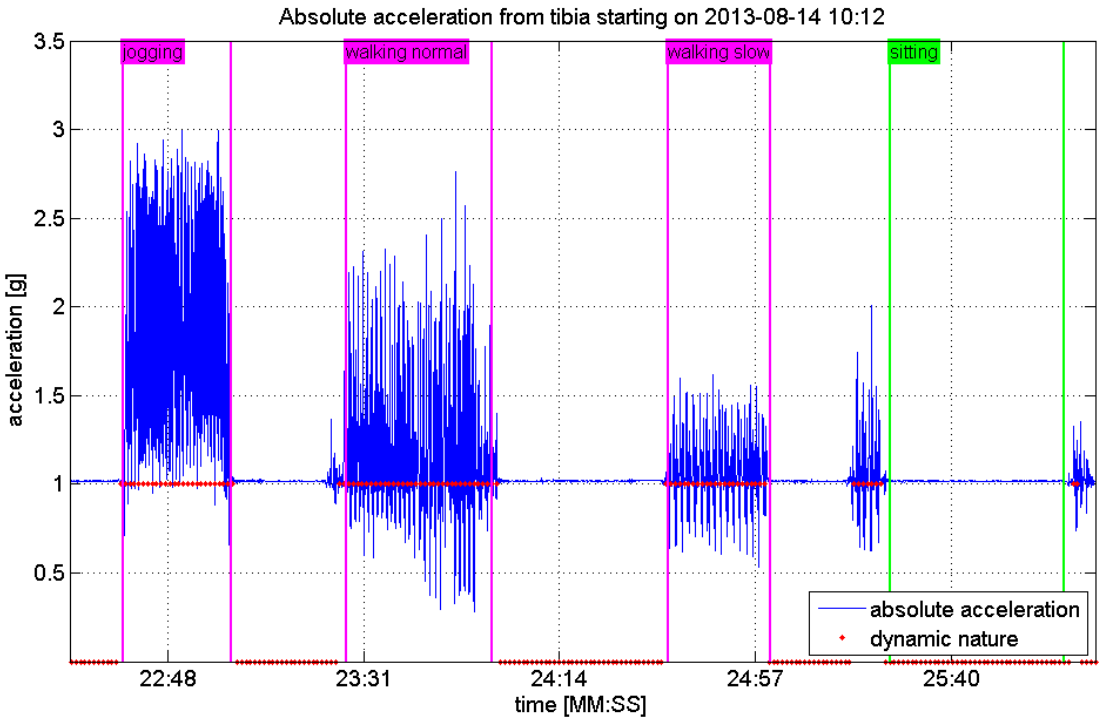


Figure 5-1: Screen for allocating during training phase with activity template

It is not obligatory to use the template for activity allocation. The template is supposed to improve the time efficiency concerning the application during the *NutriHEP* study. Single activities can be allocated by hand. Further types of locomotion, like cycling or sprinting, are not covered by the template, because these movements are prohibited during the intervention study. However, new activities can be added to the activity catalog. When naming new activities, the dynamic nature needs to be specified. If the new activity is of static nature, the number of static activities in the activity catalog (`Nstatic`) is incremented. `Nstatic` is important for the assigning of activities to the right sub-folder.

Function name: trainNN.m

This function trains an ANN by solving a pattern recognition problem. This code lines are mainly generated with the assistance of the *Neural Network Pattern Recognition Tool* `nprtool`. It is part of the *Neural Network Toolbox* in *MATLAB*.

Create a pattern recognition network

The settings of the ANNs basically are according to HANSON ET AL. [HGP+11]. The hidden layer has a size of 20 neurons. HANSON ET AL. stated that a difference of ± 5 neurons did not affect the results.

```
hiddenLayerSize = 20;
net = patternnet(hiddenLayerSize);
```

The training data set is divided randomly into training (70 %), validation (15 %) and testing (15 %) set. Scaled conjugate gradient backpropagation is used as training function and the mean square error is the performance parameter, which has to converge.

```
% Setup Division of Data for Training, Validation, Testing
net.divideFcn = 'dividerand';  % Divide data randomly
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;
net.trainFcn = 'trainscg';  % Scaled conjugate gradient
% Choose a Performance Function
net.performFcn = 'mse';  % Mean squared error
```

The ANN is trained and tested, until the mean squared error tends to a minimum. Algorithms for these procedures are provided by the *Neural Network Toolbox*.

5.4. Classification phase

5.4.1. Neural network classification

Function name: main.m

This function represents the main function for the activity classification of accelerometry data. As the classification is the main feature of the software, this function is called `main.m`. This function is customized for the analysis of two acceleration sensors (hip and tibia) as applied during the *NutriHEP* study.

The user chooses a pre-trained ANN and a folder containing acceleration data. The activity class catalog is saved within the ANN file, because new activities might have been added. If a file containing the time shift (see chapter 5.2.2) is present, the time shift is added to the corresponding data array. First, the classification for the tibia is done by calling the function `activityClassification.m`. Subsequently, the hip data classification is performed. All results are plotted and saved in a separated classification folder.

Function name: activityClassification.m

This function classifies accelerometry data into different activity classes. After the import of data and its time shift application, the data is classified by its dynamic nature. Following this, static data is classified with a static ANN and dynamic data with a dynamic ANN. Finally, all results from the classification are saved in separated matrix files for further access and applications.

Import acceleration data and classify dynamic nature

Acceleration data is imported with the `multiple_RAW_Conversion.m` function (see chapter 5.3.1). If a time shift file is present, it is applied to one of the sensors. The TXT-file contains a shift in milliseconds, which was calculated during time synchronization (see chapter 5.2.2). It is simply added to every time stamp of the acceleration data of the regarding sensor. The dynamic nature of the signal is analyzed by calling the function `dyn_natureClassification.m` (see chapter 5.3.2).

Classify static and dynamic activities separately

All static data is concatenated and the static features are extracted as described in chapter 5.3.3. The resulting static feature matrix is saved as backup and then fed to the static ANN. The same process is executed with the dynamic data.

The classification results are saved in separate matrices. In addition, the time stamps of the dynamic nature are saved as `DynamicNatureTimes.m`. This file contains the points of time, where the signal was classified as static or dynamic. This file is relevant for the compliance check in the following chapter.

5.4.2. Compliance check

Function name: complianceCheck.m

This function is used for checking the subject's compliance during the *NutriHEP* study. It compares the dynamic nature of the acceleration signal measured at hip and tibia. If a static activity is detected on the tibia, but a dynamic one on the hip, then it is considered that the subject is not wearing the tibia sensor, thus the orthosis. In all other cases all sensors are assumed to be worn. At the end, the result is plotted in a graphic.

Import classification results

The results from the classification are loaded from the file `DynamicNatureTimes.m`. It contains time stamps separated in static and dynamic activities. First, the time stamps are concatenated into one matrix for each sensor position (hip and tibia). After this, the matrices are sorted in time, which results in two matrices (hip and tibia) with time stamps and dynamic nature. Static activities are encoded as `0` and dynamic activities as `1`.

```
% load dynamic nature classifications' results
CCtibia = load(fullfile(CCfolder,'tibia','DynamicNatureTimes.mat'));
CChip = load(fullfile(CCfolder,'hip','DynamicNatureTimes.mat'));

% init and sort dynamic nature
tibiaTimes = [CCtibia.winTimesDyn,...
              CCtibia.winTimesStat;...
              ones(1,length(CCtibia.winTimesDyn)),...
              zeros(1,length(CCtibia.winTimesStat))];
hipTimes = [CChip.winTimesDyn,...
            CChip.winTimesStat; ...
            ones(1,length(CChip.winTimesDyn)),...
            zeros(1,length(CChip.winTimesStat))];
tibiaTimes = sortrows(tibiaTimes',1)';
hipTimes = sortrows(hipTimes',1)';
```

Compliance check

The compliance check looks at every sample, but 120 samples before and after every sample are taken into account. By this, only longer episodes of non-compliance are recognized. If for all samples the tibia signal is classified as static and the hip signal as dynamic, then the compliance is classified as not present, i.e. the orthosis is not worn. Compliance is binary encoded as `0` (orthosis not worn) and `1` (orthosis worn).

```
w = 120;
% if tibia is not dynamic and hip is dynamic, then tibia sensors is not worn
for i = w+1:N-w
    if ~tibiaTimes(2,i-w:i+w) == hipTimes(2,i-w:i+w)
        compliance(2,i-w:i+w) = 0;
    end
end
```

The function `plotCompl.m` shows a graphical visualization of the compliance check results. The interfaces of all plotting functions are described in the following chapter.

5.5. Graphical visualization of results

5.5.1. Classification results

Acceleration data

The end result of a classification may look like the example GUI in Figure 5-2. The acceleration signal is plotted over time on the top graph. For a better perception, the resulting absolute acceleration from all three axes is drawn and not every axis individually. The classification results are shown in the middle. A nominal scale lists the different kinds of activities. Every blue dot stands for a classification. In the bottom graph, one can see the certainty of every classification. The ANN also gives information about how sure it is about a classification. Since an ANN can only classify activities, which were trained beforehand, signals from unknown activities are stated as *unclassifiable*. This is the case, if the classification certainty is below 80 %.

Additionally, the user can zoom in and out in each window screen. All time axes are linked together, i.e. zooming in one window simultaneously changes the other windows. An external function `datetickzoom.m`[2] enables a proper zoom behavior.

[2] http://www.mathworks.com/matlabcentral/fileexchange/15029-datetickzoom-automatically-update-dateticks/content/datetickzoom.m

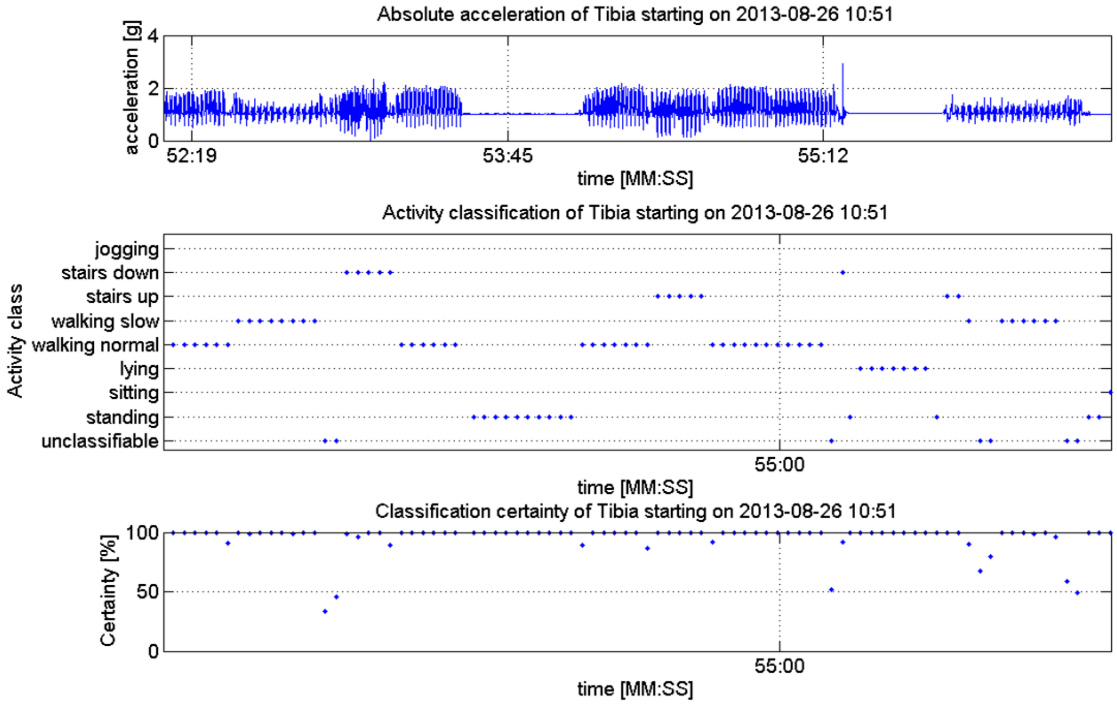


Figure 5-2: Classification result GUI

This graphical visualization of the results is mainly intended for looking at specific single classifications and the raw acceleration signal. It is plotted with the function `plotClassification.m`.

Pie chart

A visualization of the relative distribution of activities is given by a pie chart (see Figure 5-3). Its percentages are relative to the total time of classification. The occurrences of activity are counted and plotted with the built-in `pie` function.

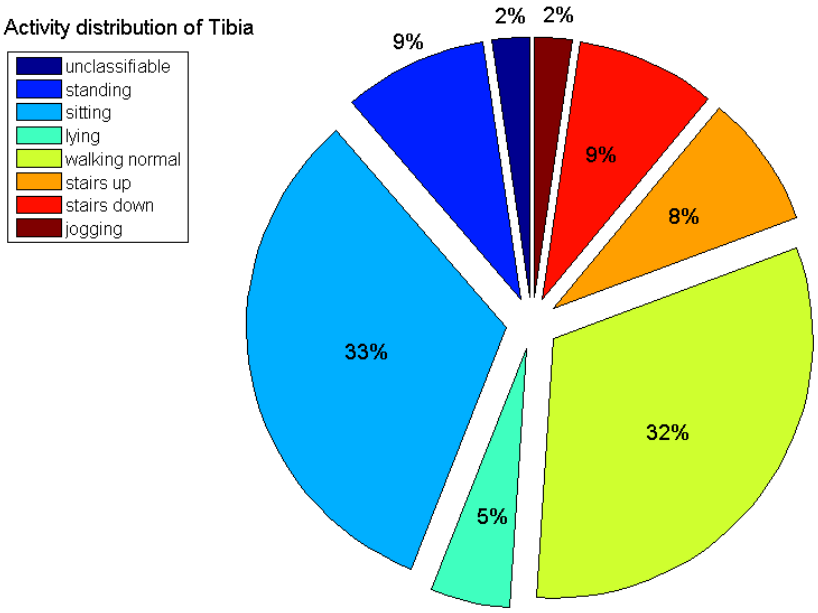


Figure 5-3: Pie chart results GUI

This graphical visualization of the results is mainly intended for looking at a qualitative activity distribution profile. It is plotted with the function `plotPie.m`.

Table

A table with numeric information is generated (see Table 5-1). At the top, it shows the elapsed time of the classified acceleration data. Beneath, the average certainties of dynamic and static classifications are written. This information helps to estimate the quality of the classification. In the lower half, all activities from the activity catalog are listed on the left. For each activity, the number of successful recognitions, its percentage share, and the estimated share of time is displayed. The calculation of the percentage share is equal to the algorithm for the pie chart. The estimated time of each activity is computed by taking the respective percentage of the total elapsed time.

Table 5-1: Table results GUI

	Days : Hours : Minutes : Seconds		
elapsed time	00 : 00 : 14 : 59.565		
	Avg. certainty of Classification [%]		
dynamic activities	99.7922		
static activities	95.7240		
all activities	97.5903		
	No. of recognitions	Activity distribution [%]	estimated Time [dd:hh:mm:ss]
unclassifiable	10	3.3003	00 : 00 : 00 : 29
standing	125	41.2541	00 : 00 : 06 : 11
sitting	6	1.9802	00 : 00 : 00 : 17
lying	8	2.6403	00 : 00 : 00 : 23
walking normal	88	29.0429	00 : 00 : 04 : 21
walking slow	9	2.9703	00 : 00 : 00 : 26
stairs up	28	9.2409	00 : 00 : 01 : 23
stairs down	21	6.9307	00 : 00 : 01 : 02
jogging	8	2.6403	00 : 00 : 00 : 23
sprinting	0	0	00 : 00 : 00 : 00
SUM	303	100	00 : 00 : 14 : 59

This graphical visualization of the results is mainly intended for looking at a quantitative activity distribution profile. It is plotted with the function `plotTable.m`.

5.5.2. Compliance results

The compliance is represented in a colored timeline. Timeframes with worn and not worn tibia sensor are encoded in green and red respectively.

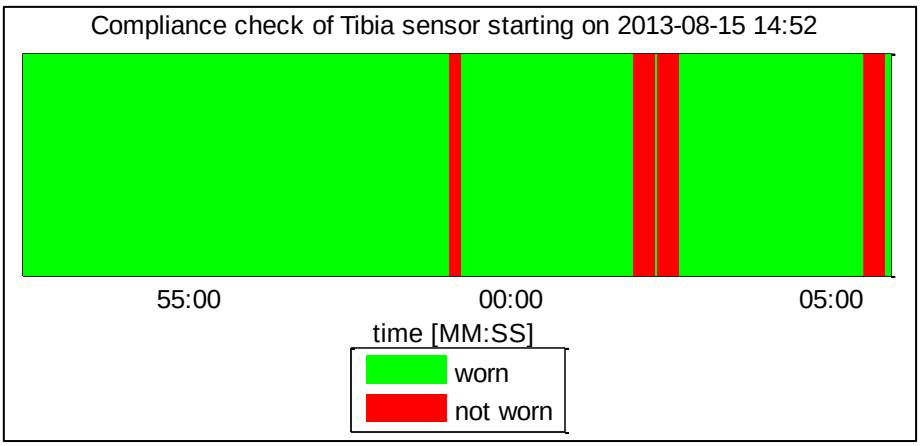


Figure 5-4: Compliance results GUI

This graphical visualization of the results is mainly intended for looking at a qualitative compliance profile. It is plotted with the function `plotCompl.m`.

5.6. Standalone executable

The *MATLAB Compiler* assists in transforming programs into standalone applications. Created applications use the *MATLAB Compiler Runtime* (MCR), which allows royalty-free deployment on computers without any valid *MATLAB* license. An installer for the MCR is packaged with the application. The standalone executable does not need any installation and can be started after copying.

The GUI of the standalone application looks like the original software program, except for the buttons for training. Unfortunately, *MATLAB Compiler* is only able to compile pre-trained network command line functions [Math13d]. I.e., training a new ANN with a standalone application is not possible. Therefore, the training functions are automatically disabled.

6. Evaluation

During the program development, algorithms were already tested with data acquired by a single person. In order to evaluate the practical suitability of the program, data of several subjects were required. With the help of this evaluation, the applicability of the program was rated for its practical use. The results of the evaluation may be used for further changes to the program.

During the work on this thesis, the conducting researcher of the *NutriHEP* study decided to integrate skin conductance sensors in the *HEP* orthosis. These sensors can detect skin contact, and thus if the orthosis is worn or not. This technology promises a high accuracy. Therefore, the requirement for the compliance check was downgraded and seen as supporting feature. Consequently, the results of the compliance check are not under consideration in this evaluation.

6.1. Trial subject recruitment

Subjects were mainly recruited from the staff of the *Department of Space Physiology*. A monetary compensation was not obtained. The number of subjects was 14 and included 7 female and 7 male subjects, so any gender variances in movement due to anatomy were considered. The average age was 29,4 ± 9,3 years (mean ± SD). For each subject, the study lasted for approximately one hour of laboratory data acquisition. Exclusion criteria were abnormalities of gait or mobility, rheumatic diseases, and pregnancy. The study contained activity measurements in the physiology lab and the building of the *Institute of Aerospace Medicine*. The subjects were not dictated to any special diet or medication.

6.2. Test procedure and course of study

The subjects wore accelerometers and ran through a given course, where several activities were performed. The first run had a predefined order and time plan and served as training data for the ANNs. The second run corresponded to the first one, but its data was intended for testing of the ANNs. The order of a third run was chosen individually by the subject, and served as test data as well.

6.2.1. Data acquisition

Main aspect of this evaluation was the acquisition of acceleration data from laboratory recordings. Accelerometers from *GCDC* were used as sensors. Subjects received two accelerometers, one on the lower leg and one on the hip. The lower leg sensor was placed at a similar position as it was during the *HEP*-study [WDM+13] (see Figure 6-1 (a)).

(a) (b)

Figure 6-1: Position of accelerometer on *HEPHAISTOS* orthosis (a) and on shin guard (b)

It was not possible to run the evaluation with subjects wearing a *HEP* orthosis, because it is a custom made orthosis. Each subject would have needed an expensive custom-built model. These costs (over 60.000 € per orthosis) were not reasonable for this evaluation. Moreover, the subjects from the past *HEP* study were not available. Instead, gear for mounting the accelerometer to the lower leg was used. A prototype was created using a shin guard as it is common in soccer sports (see Figure 6-1 (b)). The sensor was placed parallel to the tibia bone and secured with one screw to the shin guard. Adhesive tape was added to avoid a tilting of the sensor. The placement of the sensor with a shin guard was an adequate substitution to the *HEP* orthosis.

Figure 6-2: Belt bag with accelerometer

The second accelerometer was worn in a belt bag (see Figure 6-2). Thereby, the sensor was placed at the hip and closer to the center of mass of the body. It was attached with *Velcro* tape, so the sensor could not roll along its longer axis. It was checked that the sensor kept its orientation in the belt bag throughout the whole evaluation. The orientation was consistent for all subjects.

On the tibia position an *X6-2* accelerometer and on the hip position a *X6-1B* accelerometer were attached. The accelerometers were set to a sample rate of 20 samples per second, a low gain and a 12-bit resolution. How to configure the accelerometers and how to use the analyzing software is documented in the manuals of the accelerometers [GCDC10a] [GCDC10b] and the user manual of the software (see appendix 9.2).

For video documentation, a *GoPro Hero 3 Silver Edition* camera was mounted to a chest mount harness (see Figure 6-3 (b)). The camera was only used during the third run, where the subject chose an individual course. Its view was sufficient to manually identify the types of locomotion of the subjects (see Figure 6-3 (c)).

(a)

(b)

(c)

Figure 6-3: Subjects wearing tibia and hip sensor (a), *GoPro* chest mount harness (b), and *GoPro* camera point of view (c)

6.2.2. Activity classes

Subjects had to perform monitored movements in the laboratory and in the building, respectively. More precisely, a course had to be completed, which required the following activities:

Table 6-1: Activity classes

Static activities	Dynamic activities
Standing	Walking with normal pace
Sitting	Walking with slow pace
Lying	Climbing ascending stairs
	Climbing descending stairs
	Jogging with slow pace

Standing — Subject stands in upright position. Both legs are equally loaded, so that the hip remains in its horizontal position. Other movements are to be avoided.

Sitting — Subject sits on a chair. Both feet touch the ground. Other movements are to be avoided.

Lying — Subject lies on its back on a couch or table. Other movements are to be avoided.

Walking with normal pace — Subject walks across a corridor. This will be done in a uniform, everyday pace, so a 50-meter range is completed in approximately 30 seconds.

Walking with slow pace — Subject walks across physiology lab. This will be done in a uniform, slow pace, so a 20-meter range is completed in approximately 30 seconds.

Climbing ascending stairs — Subject climbs staircase several floors up. The movement should be uniform and slow-paced. No skipping steps or running allowed.

Climbing descending stairs — Subject climbs staircase several floors down. The movement should be uniform and slow-paced. Overloading or shock absorbing steps are to be avoided.

Jogging with slow pace — Subject jogs in a slow pace across a corridor. The pace should be uniform. A fast walking is to be avoided.

6.2.3. Activity course

All the activities described above were performed and recorded during an activity course (see Table 6-2). The course consisted of different stations, each demanding a certain movement. Between stations, the sensors kept recording accelerations. These movements could not be allocated precisely to an activity, but still needed to be protocoled as interstation activities.

For better differentiation of the activities in the visual representation, it was important to start and end dynamic activities with static ones and vice versa. According to the durations stated in Table 6-2, one run through the course took less than 15 minutes. Every subject had to complete the course twice. Afterwards, the subject was ordered to do a third individual run. The subject was free to go around the building, moving with any type of locomotion. The only requirement was that every activity from Table 6-1 was performed for at least 10 seconds throughout the entire run. Figure 6-4 shows a map of the *DLR* campus and where the different activity stations were located.

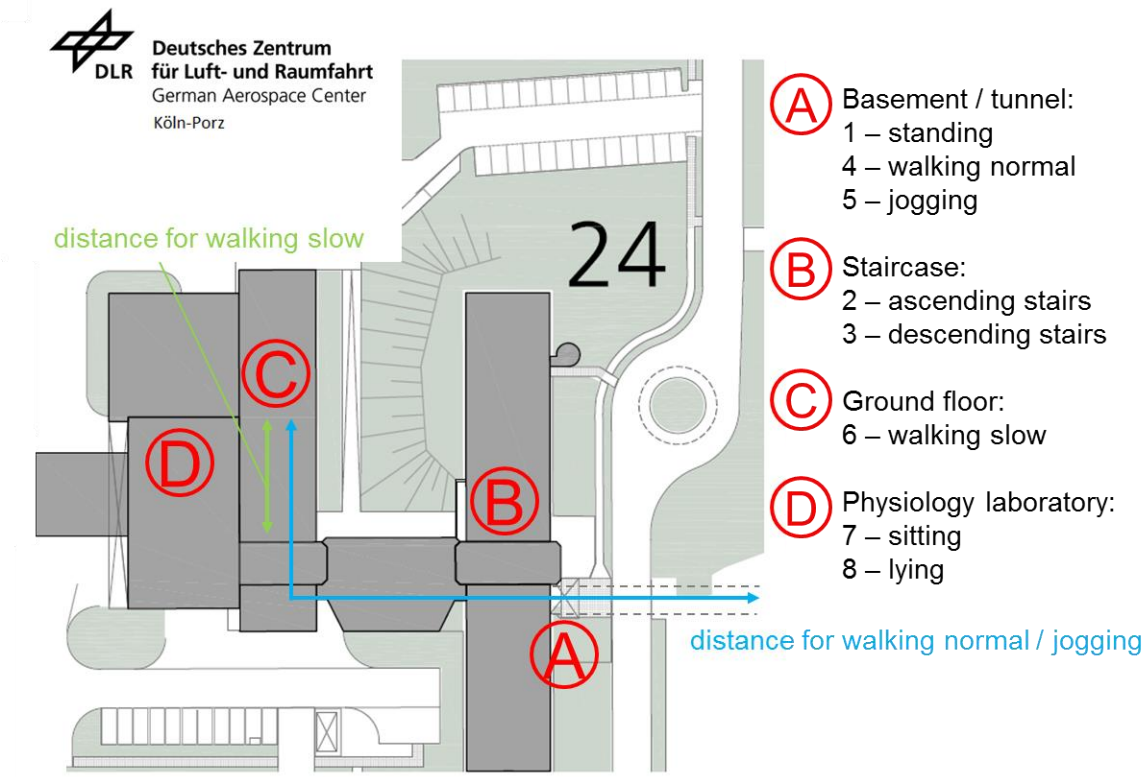


Figure 6-4: Map of *DLR* campus with building 24 and locations of the stations

Risk-benefit analysis and precautions

Since there is no medical intervention, the only risk lies in the completion of the activity course. Its demanding activities are elemental types of locomotion of everyday life, thus, not containing any serious risks. Therefore, no medical advisor or paramedic is needed during the evaluation.

Table 6-2: Order of activity course

No.	Starting time	Activity	Description	Duration
1	0:00	*Standing*	Turn accelerometers on, stand in front of stairs, and wait for 0:30	30 s
2	0:30	*Climbing ascending stairs*	Walk 4 floors upstairs in a uniform pace	~ 90 s
		Interstation activity	Stand still at top level, turn around, and wait for 2:30	~ 30 s
3	2:30	*Climbing descending stairs*	Walk 4 floors downstairs without skipping or jumping	~ 90 s
		Interstation activity	Stop at end of stairs, go to next station, and wait for 5:00	~ 60 s
4	5:00	*Walking with normal pace*	Walk in own normal pace	~ 90 s
		Interstation activity	Stop at end of corridor, turn around, and wait for 7:00	~ 30 s
4	7:00	*Walking with normal pace*	Walk back in own normal pace	~ 90 s
		Interstation activity	Stop at end of corridor, go to next station, and wait for 9:30	~ 60 s
5	9:30	*Running as jogging*	Jog in own slow pace	~ 30 s
		Interstation activity	Stop at end of corridor, turn around, and wait for 11:30	~ 90 s
6	11:30	*Walking with slow pace*	Walk slowly across corridor	~ 30 s
		Interstation activity	Stop at end of corridor, go to next station, and wait for 13:00	~ 60 s
7	13:00	*Sitting*	Sit down on chair with both feet touching ground	30 s
		Interstation activity	Get up from chair, go to next station, and wait for 14:00	~ 30 s
8	14:00	*Lying*	Lie down on couch with face up and without crossing legs	30 s
		Interstation activity	Get up from couch turn accelerometers off	~ 30 s

6.3. Test data processing

6.3.1. Training and classification

With the acceleration data from the first run the ANNs were trained and the resulting network data saved. How to use the training feature of the software is explained in its user manual (see appendix 9.2). For each subject, an individual ANN was trained. The template feature helped to save time while training 14 different ANNs. The data from the second and third run were classified by the analyzing feature. Its results were then sighted for evaluation.

6.3.2. Evaluation results

Randomization

The first and the second run were identical in their sequence and therefore interchangeable. One served as training set, the other one was for classification. However, doing the activity course twice, might have had an effect on the locomotion. It was observed that some subjects were excited during the first run, because the course was new and unknown. It cannot be ruled out that the subject paid more attention to its movements under the experimental conditions. Additionally, the second run seemed to be boring, because of its redundancy. The subject might have paid less attention and moved in a more ordinary way. To eliminate these possible effects, a randomization was applied, whether the first or the second run served as training set. Further, the position of the tibia sensor was randomized. Five subjects wore the shin guard on the left leg, the rest on the right leg.

For a better differentiation, the classification of the first or second run is referred to as *activity course* (AC) classification. The classification of the third run is referred to as *free run* (FR) classification.

Evaluation results

The results of a classification were compared to the actual activities performed by the subject. For the second run, the order was given in Table 6-2. For the third run, the results had to be compared to the video recording. An example of a classification result is shown in Figure 6-5.

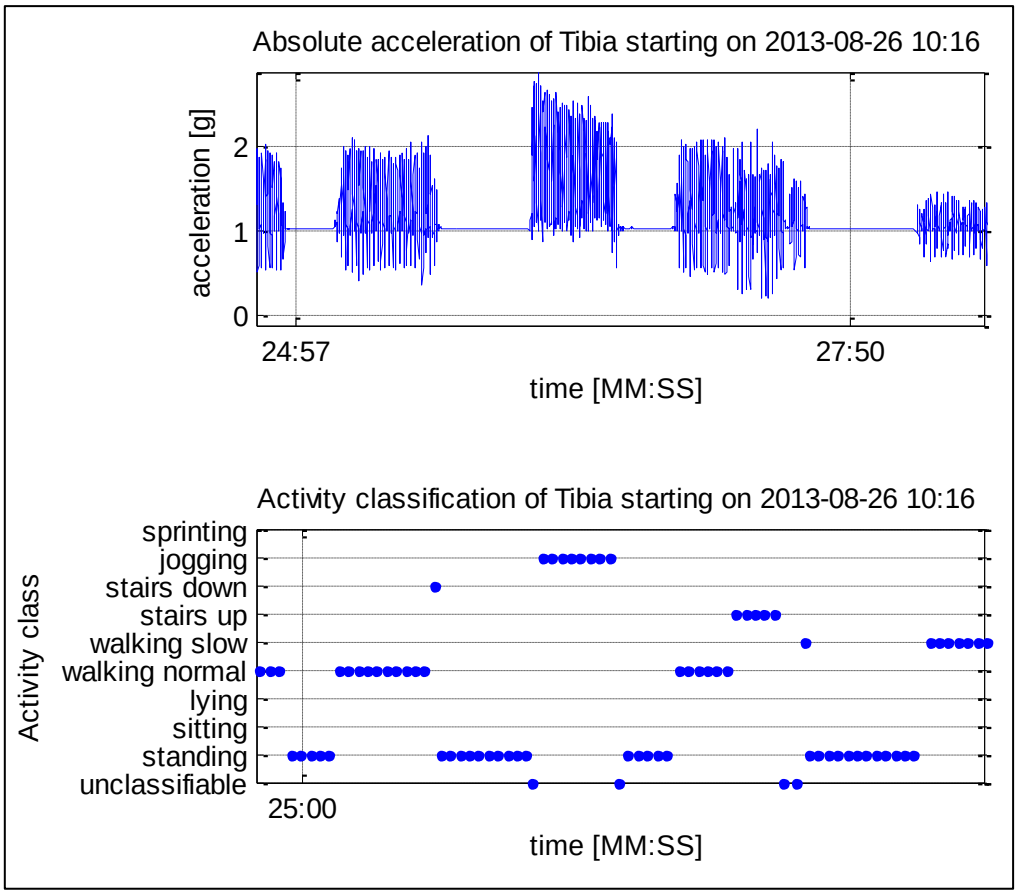


Figure 6-5: Classification result example of a 2nd run

The acceleration signal is plotted in the upper part. Below, the results of activity classification are shown. Every blue dot stands for a classification of the activity according to the axis of ordinates. Unsuccessful classifications are defined as *unclassifiable*. According to Table 6-2, it is known which activity was performed by the subject. True activity and classification result are comparable. Therefore, the numbers of correct and incorrect classifications can be obtained.

To evaluate a classifying ANN system, a confusion matrix was created. How to fill out a confusion matrix and how to quantify performance parameters was described in chapter 3.5. The numbers of successful and unsuccessful classifications were obtained by counting the blue dots in the classification results graphic (see Figure 6-5).

The creation of the confusion matrix for an AC classification was an unambiguous task. When and which movements were performed was clearly specified by the order of the activity course (see Table 6-2). In contrast, interstation activities were ignored.

Evaluating the FR classification results was more difficult. Type and order of movements were not specified in the third run, but instead had to be interpreted by video sighting. This implied a subjective interpretation of types of locomotion. The diffi-

culty was to judge unclassifiable movements performed by the subjects. Most common was the case of transition from one movement to another, which was often classified as *unclassifiable*. It seemed correct to evaluate these classifications as successful. Longer episodes of one movement were clearly identifiable in the video. Movements, which were not listed in the activity list in Table 6-1, were allocated as *unclassifiable* activities.

For every subject there were 4 confusion matrices created. Two for the AC classification (with hip and tibia sensor) and two for the free run classification (with hip and tibia sensor). This leads to a total number of 56 confusion matrices, 28 of AC and 28 of FR classification. To evaluate the recognition software with its ANN as classifying system, accuracies and reliability of AC and FR classification were obtained by merging all subjects' results.

6.4. Description of results

Table 6-4 shows a summarizing confusion matrix for the hip sensor of the AC classifications from all subjects. In addition to the absolute number of classifications, each cell contains the percentage relative to the total number of classifications (i.e. in Table 6-4 in the first cell, 129 correct classifications of *Standing* represent 7,13 % of all classifications). The highest occurrences of misclassifications (off-diagonal elements) were in the *unclassifiable* column. This means that the classification certainty was not high enough, but it only affected the accuracy of the classifier. However, the most activity-activity misclassifications were between *Sitting* and *Standing*. This indicates difficulties for the classifier distinguishing these two activities. Looking at the accuracies on the right column, the lowest producer's accuracy was measured for the activity *Walking slow*. A higher reliability of 89,2 % indicates that the classifier is able to distinguish such locomotion from others. The remaining reliabilities are over 80 %.

The same descriptions are applicable to the results of the tibia sensor during the AC classification in Table 6-5. With both feet on the ground while sitting, the orientation does not change during standing. This explains the lower accuracies for *Sitting* and *Standing*. Again, the accuracy for the activity *Walking slow* is the lowest.

Activity course classification with hip and tibia sensor is accurate and reliable for almost every of the activities stated in Table 6-1. The activities *Standing* and *Sitting* are more often confused than others. Half of the activity *Walking slow* is not identified, but nearly 90 % of *Walking slow* classifications is correct.

Such a confusion matrix as in Table 6-4 is not suitable for the free run classification. Time spent per activity varies between subjects, because every run was chosen

individually. This uneven distribution would distort the confusion matrix and is considered as inexpedient.

The evaluation with confusion matrices showed that the accuracy of the activity course classification lied above 85 % and its results were in over 95 % of the cases reliable (see Table 6-3). This was true for hip and tibia classification of the data set based on the completion of the activity course. For the classification of the free run accuracy and reliability were over 70 % and 75 %, respectively.

Table 6-3: Accuracies and reliability of classifier (mean ± SD)

	Overall accuracy [%]	Average reliability [%]
Activity course	87,99 ± 7,54	96,49 ± 3,16
Free run	71,23 ± 16,55	76,77 ± 16,26

An approximate computation time of 6 minutes per measurement day was measured. This is true for acceleration acquisitions with a sample rate of 20 Hz.

Confusion matrix

Actual class \ Classification result	Standing	Sitting	Lying	Walking normal	Walking slow	Stairs up	Stairs down	Jogging	unclassi-fiable	Accuracy
Standing	129 *7,13%*	5 *0,28%*	1 *0,06%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	12 *0,66%*	87,8%
Sitting	19 *1,05%*	84 4,64%	5 *0,28%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	5 *0,28%*	74,3%
Lying	0 *0,00%*	0 *0,00%*	108 5,97%	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	16 *0,88%*	87,1%
Walking normal	0 *0,00%*	0 *0,00%*	0 *0,00%*	644 35,60%	7 *0,39%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	27 *1,49%*	95,0%
Walking slow	2 *0,11%*	0 *0,00%*	11 *0,61%*	4 *0,22%*	58 3,21%	2 *0,11%*	0 *0,00%*	0 *0,00%*	34 *1,88%*	52,3%
Stairs up	0 *0,00%*	0 *0,00%*	0 *0,00%*	1 *0,06%*	0 *0,00%*	259 14,32%	0 *0,00%*	0 *0,00%*	23 *1,27%*	91,5%
Stairs down	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	5 *0,28%*	212 11,72%	0 *0,00%*	30 *1,66%*	85,8%
Jogging	0 *0,00%*	0 *0,00%*	0 *0,00%*	9 *0,50%*	0 *0,00%*	0 *0,00%*	1 *0,06%*	92 5,09%	4 *0,22%*	86,8%
unclassifiable	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 0,00%	no data
Reliability	86,0%	94,4%	86,4%	97,9%	89,2%	97,4%	99,5%	100,0%	0,0%	**82,57%** Average Accuracy
										93,85% Average Reliability
										87,67% Overall Accuracy

Table 6-4: AC confusion matrix from hip

Confusion matrix

Actual class \ Classification result	Standing	Sitting	Lying	Walking normal	Walking slow	Stairs up	Stairs down	Jogging	unclassifiable	Accuracy
Standing	106 *6,06%*	17 *0,97%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	17 *0,97%*	75,7%
Sitting	15 *0,86%*	81 4,63%	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	11 *0,63%*	75,7%
Lying	1 *0,06%*	0 *0,00%*	107 6,12%	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	3 *0,17%*	96,4%
Walking normal	0 *0,00%*	0 *0,00%*	0 *0,00%*	633 36,19%	0 *0,00%*	0 *0,00%*	1 *0,06%*	0 *0,00%*	30 *1,72%*	95,3%
Walking slow	9 *0,51%*	0 *0,00%*	0 *0,00%*	7 *0,40%*	61 3,49%	1 *0,06%*	4 *0,23%*	0 *0,00%*	29 *1,66%*	55,0%
Stairs up	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	268 15,32%	0 *0,00%*	0 *0,00%*	10 *0,57%*	96,4%
Stairs down	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	203 11,61%	0 *0,00%*	37 *2,12%*	84,6%
Jogging	0 *0,00%*	0 *0,00%*	0 *0,00%*	8 *0,46%*	0 *0,00%*	0 *0,00%*	1 *0,06%*	89 5,09%	0 *0,00%*	90,8%
unclassifiable	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 *0,00%*	0 0,00%	no data
Reliability	80,9%	82,7%	100,0%	97,7%	100,0%	99,6%	97,1%	100,0%	0,0%	**83,74%** Average Accuracy **94,75%** Average Reliability **88,51%** Overall Accuracy

Table 6-5: AC confusion matrix from tibia

7. Discussion and conclusion

7.1. Discussion

Discussion of evaluation results

In order to evaluate the developed classifier, confusion matrices of classification results were created. These matrices give some indication of misclassifications and performance parameters.

Most activity-activity misclassifications occurred with *Sitting* and *Standing*. This indicates difficulties for the classifier to distinguish between these two activities. This confusion seems reasonable, because the position of the hip may not change noticeably. Since *Sitting* and *Standing* are static activities, the orientation of the accelerometer is a decisive factor for the classification.

The lowest producer's accuracy was measured for the activity *Walking slow*. Apparently, walking slowly is a type of locomotion where the acceleration patterns vary substantially. It seems that the instructions for walking slowly mast have been vague so the subjects were unable to repeat this locomotion in the same manner. Furthermore, the walking behavior may differ in various surroundings. *Walking slow* may be performed differently on a big parking lot than in a small corridor. However, a higher reliability of *Walking slow* indicates that the classifier is able to distinguish it from other locomotion. The remaining reliabilities exceed 80 %, which is a satisfying result. The same conclusions are applicable to the results of the tibia sensor during the AC classification.

Looking at the performance parameters of the classifier, it seems consistent that accuracy and reliability were high for the AC classification, because environment, pace and motivation were identical for training and test data set. Nevertheless, this fact does not prove the proper functionality of the analyzing algorithm in the recognition software. The free run simulates more practical conditions and can therefore be regarded as more representative and its evaluation results as more suitable. However, the free run performance parameters showed a higher SD and more outliers. A free run without any protocol allowed the subjects to do any kind of locomotion. Some subjects wanted to put the software on trial by moving in unusual ways (e.g. climbing stairs backwards or doing sit ups). This led to a higher number of *unclassifiable* activities in relation to the total acquisition, which led to a lower accuracy and reliability.

The classifying system in this thesis has a similar structure like the classifier by LEE ET AL. [LKK+10] and its accuracy parameters are similar as well. However, neither LEE ET AL. nor other authors published any statement about reliability of a classifier.

The classifier developed in this thesis is able to correctly predict new and unknown data with a sufficient accuracy and reliability. The speed of the classification is decent. Training for each subject is needed beforehand, but the resulting ANN is applicable for the rest of the study. An approximate computation time of 6 minutes per measurement day is considered adequate, because there is no mention of any minimum computation time in literature.

The compliance check showed reasonable results, but its accuracy needs to be proven in a further evaluation. During the work on this thesis, the conducting researcher of the *NutriHEP* study decided to integrate skin conductance sensors in the *HEP* orthosis. These sensors can detect skin contact, and thus if the orthosis is worn or not. This technology promises a high accuracy. Therefore, the requirement for the compliance check was downgraded and seen as supporting feature.

Discussion of usability test results

In the course of this thesis, only a short usability test was carried out with the conducting researcher of the *NutriHEP* study. This test involved studying the user manual, fulfilling all possible tasks, and completing a subsequent questionnaire. This evaluation of usability by only one person was considered acceptable, because the focus of this thesis was not on the development of a marketable software package.

In the usability test, the program software and its use were experienced comprehensible and efficient. The tester commented that at some points the program did not explain itself so that necessarily the user manual had to be consulted. This point of criticism may be justified on grounds of user-friendliness, but since a user interface overloaded with information leads to confusion, the information missing from the interface were deliberately placed in the user manual.

The user manual itself was considered helpful and very clearly written. Especially figures of screen shots were appreciated. Some step-by-step instructions skipped certain steps and led to confusion, which is why the user manual was not valued as fully understandable. Missing steps were pointed out by the conducting researcher and were corrected appropriately.

The diagrams which present the results of classifications also received a satisfactory validation. Neither is there any superfluous diagram nor is any unnecessary information shown. Furthermore, the diagrams allow a simple and quick overview of the results and are therefore very likely to be used in future research work.

In short, it can be said that the usability of the software program supports the researcher in her studies and simplifies the work flow.

7.2. Conclusion

In this work, a software program for human activity pattern recognition was successfully developed. With the help of trained ANNs a classification of different activities was possible. The ANNs were trained with supervised learning algorithms and the required ground truth data was retrieved from activity course acquisitions. A combination of features from the time domain and frequency domain were extracted to describe the acceleration signal. Different graphical visualizations represent the classification results.

Table 7-1: Requirements and their accomplishments

	Requirements	accomplished through	
Req1	*"When did the subject do which body movement?"*	Overview graph of daily physical activity	☑
Req2	*"How long did the subject do which body movement?"*	Overview graph of daily physical activity	☑
Req3	*"Was the orthosis worn the entire time?"*	Overview of compliance	☑*
Req4	*"What does the data look like during unidentified activities?"*	Overview graph of daily physical activity	☑
Req5	*"The software should be easy to use."*	Usability test	☑
Req6	*"The software should work as a standalone executable."*	Standalone application	☑
Req7	*"The software should be provided in English and German."*	Multilingual implementation	☑
Req8	*"The software should not be time consuming."*	Usability test	☑
Req9	*"The software should work with a single sensor on any position."*	Successful implementation	☑
Req10	*"The software should work with GCDC accelerometers."*	Use of GCDC accelerometers	☑

* Compliance can be checked, but the results are unevaluated

In chapter 2, numerous user requirements were collected, which have been all accomplished through one software program (see Table 7-1). Diagrams showing the classified activity over any given time can be generated (Req1). The user can also have a look at the raw acceleration data and review unidentified activities (Req4). Pie charts show the distribution of activities (Req2). A compliance check can be executed to see if the orthosis was worn or not (Req3). The ease of use of the GUI was tested

and approved in a usability test (Req5). It offers an English and German language package (Req7) and was valued as time efficient (Req8). Furthermore, it is possible to work with the software program on any computer system without *MATLAB* license as a standalone executable (Req6). In addition, the software program is usable beyond the *NutriHEP* study and allows activity analysis with a single accelerometer from *Gulf Coast Data Concepts* (Req9 and Req10). Accuracies of 87,99 % and 71,23 % were approached in classifying the activity protocol and the free run, respectively. Consequently, the required accuracy of more than 80 % is only partly approached, but still considered as entirely sufficient (SysReq5).

7.3. Outlook

Further experimental investigations will be needed to estimate the suitability of the selected features in this work. A principal component analysis may identify the most prominent features and allows reducing the number of features. Fewer features can reduce the computational complexity and remove possible redundancy in features without impairing the classification results. Additionally, other signal describing methods may be added to the pool of features to improve the performance of the classifier.

The evaluation in this thesis is based on a limited choice of activities. The activity catalog represents everyday activities which are assumed to be performed frequently during the *NutriHEP* study. New activities (e.g. cycling, driving, or working) may be added to the activity catalog during the evaluation. Furthermore, a training and evaluation set of acceleration data in a more practical environment may have enhancing effects. A subject could be sent home for one or two days wearing a camera mounted to the chest. The activities recorded by the camera are no longer performed under laboratory conditions. This may lead to better reliability and accuracy parameters.

A further issue to resolve for future work is the classification of static activities. The demonstrated classifier has problems with distinguishing between sitting and standing. This is acceptable for the use during the *NutriHEP* study because sitting and standing have similar effects on the leg muscles. However, future use may require a distinct differentiation. Further work on feature selection and accelerometer positioning would help resolving this problem.

Concerning the accelerometer position, sensor displacement is an important issue for future research. Since there is a chance that the sensor orientation may shift and attachments may loosen, enhanced sensor attachments are needed or a method for orientation independent acquisition must be developed.

For a wider use of the software program, future work should enhance the adjustability of the classifier. Most algorithm variables are hardcoded and can only be changed by a programmer. Instead, a few variables should be adjustable by the user via the GUI. These variables may involve the number of hidden neurons, window sizes, overlap percentage, and activity template.

This master's thesis presents a human activity pattern recognition system, which will be used in the *NutriHEP* study. The software program will monitor the activity distribution of a subject and check that the orthosis was worn. A further possible application is during bed rest studies, where it would be interesting to examine a subject's activity profile before and after bed rest. Moreover the developed software program may be useful in the field of ambient assisted living, where activity recognition is an important issue for the situation adaptability of home care systems.

8. References

[AgBe00] S. Agatonovic-Kustrin and R. Beresford, “Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research,” *Journal of Pharmaceutical and Biomedical Analysis*, vol. 22, no. 5, pp. 717–727, 2000.

[Alve07] T. M. Alves, “Spektrale Entropie und Bispektral Index als Messgrößen für die Wirkung von Propofol auf das EEG,” Universitäts- und Landesbibliothek Bonn, 2007.

[BaHa00] I. A. Basheer and M. Hajmeer, “Artificial neural networks: fundamentals, computing, design, and application,” *Journal of Microbiological Methods*, vol. 43, no. 1, pp. 3–31, 2000.

[Beuc11] O. Beucher, “LTI-Systeme und Stochastische Signale,” in *Signale und Systeme: Theorie, Simulation, Anwendung*, Berlin, Heidelberg: Springer, pp. 399–542, 2011.

[BKV+97] C. V. Bouten, K. T. Koekkoek, M. Verduin, R. Kodde, and J. D. Janssen, “A triaxial accelerometer and portable data processing unit for the assessment of daily physical activity,” *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 3, pp. 136–147, 1997.

[BMSG80] A. Bhattacharya, E. P. McCutcheon, E. Shvartz, and J. E. Greenleaf, “Body acceleration distribution and O2 uptake in humans during running and jumping,” *Journal of Applied Physiology*, vol. 49, no. 5, pp. 881–887, 1980.

[Böhm03] C. Böhm, “Knowledge Discovery in Databases,” Lecture, Ludwig-Maximilians-Universität München, Munich, 2003.

[ChPS08] W.-Y. Chung, A. Purwar, and A. Sharma, “Frequency domain approach for activity classification using accelerometer,” in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pp. 1120–1123, 2008.

[Cimi07] P. Cimiano, “Maschinelles Lernen,” Lecture, Universität Heidelberg, Heidelberg, 2007.

[CoTu65] J. W. Cooley and J. W. Tukey, “An Algorithm for the Machine Calculation of Complex Fourier Series,” *Mathematics of Computation*, vol. 19, no. 90, pp. 297–301, 1965.

[DeSS11] C. Demant, B. Streicher-Abel, and A. Springhoff, “Querschnitt: Klassifikation,” in *Industrielle Bildverarbeitung*, Berlin, Heidelberg: Springer, pp. 171–194, 2011.

[Dobr09] A. Dobra, “Decision Tree Classification,” in *Encyclopedia of Database Systems*, New York: Springer, pp. 765–769, 2009.

[Dorf13] G. Dorffner, “Klinische Signalverarbeitung und Mustererkennung,” Lecture, Medizinische Universtität Wien, Vienna, 2013.

[DSD+12] K. Deere, A. Sayers, G. Davey Smith, J. Rittweger, and J. H. Tobias, “High impact activity is related to lean but not fat mass: findings from a population-based study in adolescents,” *International Journal of Epidemiology*, vol. 41, no. 4, pp. 1124–1131, 2012.

[DuVe90] P. Duhamel and M. Vetterli, “Fast Fourier transforms: a tutorial review and a state of the art,” *Signal processing*, vol. 19, no. 4, pp. 259–299, 1990.

[ErPC08] M. Ermes, J. Pärkkä, and L. Cluitmans, “Advancing from offline to online activity recognition with wearable sensors,” in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pp. 4451–4454, 2008.

[FDFC10] D. Figo, P. C. Diniz, D. R. Ferreira, and J. M. Cardoso, “Preprocessing techniques for context recognition from accelerometer data,” *Personal and Ubiquitous Computing*, vol. 14, no. 7, pp. 645–662, 2010.

[FNRA10] K. Frank, M. J. V. Nadales, P. Robertson, and M. Angermann, “Reliable real-time recognition of motion related human activities using MEMS inertial sensors,” in *Proceedings of the 23rd International Technical Meeting of the Satellite Division of the Institute of Navigation*, pp. 2906–2912, 2010.

[FrJo98] M. Frigo and S. G. Johnson, “FFTW: An adaptive software architecture for the FFT,” in*. Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 3, pp. 1381–1384, 1998.

[GCDC10a] Gulf Coast Data Concepts, LLC, “GCDC X6-1A User Manual,” Gulf Coast Data Concepts, Waveland, MS, 2010.

[GCDC10b] Gulf Coast Data Concepts, LLC, “GCDC X6-2 User Manual,” Coast Data Concepts, Waveland, MS, 2010.

[GCDC13] A. Kooney, Gulf Coast Data Concepts, “Re: [X6-2 Accelerometer] Physical principle”, Email, 05-Jul-2013.

[GeMC06] K. O. Genc, V. E. Mandes, and P. R. Cavanagh, “Gravity replacement during running in simulated microgravity,” *Aviation, Space, and Environmental Medicine*, vol. 77, no. 11, pp. 1117–1124, 2006.

[GjGC10] H. Gjoreski, M. Gams, and I. Chorbev, “3-axial accelerometers activity recognition,” *ICT Innovations*, pp. 51–58, 2010.

[HeKi08] A. Hein and T. Kirste, “Towards Recognizing Abstract Activities: An Unsupervised Approach,” in *Proceedings of the 2nd Workshop on Behaviour Monitoring and Interpretation, BMI 2008*, Kaiserslautern, vol. 8, pp. 102–114, 2008.

[HGP+11] A. M. Hanson, K. M. Gilkey, G. P. Perusek, D. A. Thorndike, G. A. Kutnick, C. M. Grodsinsky, A. J. Rice, and P. R. Cavanagh, “Miniaturized Sensors to Monitor Simulated Lunar Locomotion,” *Aviation, Space, and Environmental Medicine*, vol. 82, no. 2, pp. 128–132, 2011.

[HrLe05] D. Hristu-Varsakelis and W. S. Levine, *Handbook of networked and embedded control systems*. Boston: Birkhäuser, 2005.

[ITCI13] ITC ILWIS, “Confusion matrix.” [Online]. Available: http://spatial-analyst.net/ILWIS/htm/ilwismen/confusion_matrix.htm. [Accessed: 24-Oct-2013].

[Karr10] U. Karrenberg, *Signale-Prozesse-Systeme: eine multimediale und interaktive Einführung in die Signalverarbeitung*. Heidelberg: Springer, 2010.

[KhLK08] A. M. Khan, Y.-K. Lee, and T.-S. Kim, “Accelerometer signal-based human activity recognition using augmented autoregressive model coefficients and artificial neural nets,” in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pp. 5172–5175, 2008.

[KLLK10] A. M. Khan, Y.-K. Lee, S. Lee, and T.-S. Kim, “Accelerometer’s position independent physical activity recognition system for long-term activity monitoring in the elderly,” *Medical and Biological Engineering and Computing*, vol. 48, no. 12, pp. 1271–1279, 2010.

[KNEK08] T. Van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, “Accurate activity recognition in a home setting,” in *Proceedings of the 10th International Conference on Ubiquitous Computing*, pp. 1–9, 2008.

[Kram09] O. Kramer, *Computational Intelligence - Eine Einführung*. Berlin, Heidelberg: Springer, 2009.

[Kraw13] M. Krawczak, *Multilayer Neural Networks - A Generalized Net Perspective*. Cham: Springer, 2013.

[KwWM11] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, “Activity recognition using cell phone accelerometers,” *ACM SIGKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[LKK+10] M.-W. Lee, A. M. Khan, J.-H. Kim, Y.-S. Cho, and T.-S. Kim, “A single tri-axial accelerometer-based real-time personal life log system capable of activity classification and exercise information generation,” in *Engineering in Medicine and Biology Society, 2010. EMBS 2010. 32nd Annual International Conference of the IEEE*, pp. 1390–1393, 2010.

[LoYA09] X. Long, B. Yin, and R. M. Aarts, “Single-accelerometer-based daily physical activity classification,” in *Engineering in Medicine and Biology Society, 2009. EMBS 2009. 31st Annual International Conference of the IEEE*, pp. 6107–6110, 2009.

[Math13] The MathWorks, Inc., “Fast Fourier transform - MATLAB fft,” *Mathworks*. [Online]. Available: `http://www.mathworks.com/help/matlab/ref/fft.html`. [Accessed: 05-Oct-2013].

[Math13a] The MathWorks, Inc., “MATLAB - Using FFT Demo,” *Mathworks*. [Online]. Available: `http://www.mathworks.com/help/matlab/examples/using-fft.html`. [Accessed: 06-Oct-2013].

[Math13b] The MathWorks, Inc., “Power spectral density - MATLAB dspdata.psd,” *Mathworks*. [Online]. Available: `http://www.mathworks.com/help/signal/ref/dspdata.psd.html`. [Accessed: 07-Oct-2013].

[Math13c] The MathWorks, Inc., “Autoregressive (AR) all-pole model parameters estimated using Burg method - MATLAB arburg - MathWorks Deutschland,” *Mathworks*. [Online]. Available: `http://www.mathworks.de/de/help/signal/ref/arburg.html`. [Accessed: 05-Nov-2013].

[Math13d] The MathWorks, Inc., “MATLAB Compiler 5.0 Support for MATLAB and Toolboxes - MathWorks Deutschland,” *MATLAB Compiler*, 2013. [Online]. Available: `http://www.mathworks.de/products/compiler/supported/compiler_support.html`. [Accessed: 20-Nov-2013].

[MCLC04] M. J. Mathie, A. C. Coster, N. H. Lovell, and B. G. Celler, “Accelerometry: providing an integrated, practical method for long-term, ambulatory monitoring of human movement,” *Physiological Measurement*, vol. 25, no. 2, p. R1, 2004.

[Müll13] D. Müller-Wichards, “Diskrete und schnelle Fourier-Transformation,” in *Transformationen und Signale*, Wiesbaden: Springer, pp. 161–199, 2013.

[Neub12] A. Neubauer, “Schnelle Fourier-Transformation,” in D*FT - Diskrete Fourier-Transformation*, Vieweg + Teubner Verlag, pp. 125–139, 2012.

[OpSc10] A. V. Oppenheim and R. W. Schafer, *Discrete-time signal processing*. Upper Saddle River: Pearson, 2010.

[PGKH09] S. J. Preece, J. Y. Goulermas, L. P. J. Kenney, and D. Howard, “A Comparison of Feature Extraction Methods for the Classification of Dynamic Activities From Accelerometer Data,” *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 3, pp. 871–879, 2009.

[Prus06] H. Pruscha, “Clusteranalyse,” in *Statistisches Methodenbuch - Verfahren, Fallstudien, Programmcodes*, Berlin, Heidelberg: Springer, pp. 289–316, 2006.

[RDML05] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, “Activity recognition from accelerometer data,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 20, p. 1541, 2005.

[SaWe11] C. Sammut and G. I. Webb, *Encyclopedia of Machine Learning*. New York: Springer, 2011.

[SENS08] SENSR, “Practical Guide To Accelerometers,” *Practical Guide To Accelerometers*. [Online]. Available: `http://www.sensr.com/pdf/practical-guide-to-accelerometers.pdf`. [Accessed: 08-Nov-2013].

[ShWe64] C. E. Shannon and W. Weaver, The *Mathematical Theory of Communication*. Urbana: The University of Illinois Press, 1964.

[Smit03] S. W. Smith, *Digital signal processing: a practical guide for engineers and scientists*. San Diego, CA: California Technical Pub., 2003.

[Ting10] K. M. Ting, “Confusion Matrix,” in *Encyclopedia of Machine Learning*, New York: Springer, p. 209, 2010.

[Tu96] J. V. Tu, “Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes,” *Journal of Clinical Epidemiology*, vol. 49, no. 11, pp. 1225–1231, 1996.

[Wags02] K. L. Wagstaff, “Intelligent clustering with instance-level constraints,” Cornell University, Ithaca, NY, 2002.

[WDM+13] T. Weber, M. Ducos, E. Mulder, F. Herrera, G.-P. Brüggemann, W. Bloch, and J. Rittweger, “The specific role of gravitational accelerations for arterial adaptations,” *Journal of Applied Physiology*, vol. 114, no. 3, pp. 387–393, 2013.

[WDY+13] T. Weber, M. Ducos, P. Yang, D. Jos, P. Frings-Meuthen, G.-P. Brüggemann, W. Bloch, and J. Rittweger, “The HEPHAISTOS study: Compliance and adherence with a novel orthotic device for calf muscle unloading,” *Journal of Musculoskeletal and Neuronal Interactions*, [to be published] 2013.

[WSW+13] S. Weber-Carstens, J. Schneider, T. Wollersheim, A. Assmann, J. Bierbrauer, A. Marg, H. Al Hasani, A. Chadt, K. Wenzel, and S. Koch, “Critical Illness Myopathy and GLUT4: Significance of Insulin and Muscle Contraction,” *American Journal of Respiratory and Critical Care Medicine*, vol. 187, no. 4, pp. 387–396, 2013.

[YaHs10] C.-C. Yang and Y.-L. Hsu, “A Review of Accelerometry-Based Wearable Motion Detectors for Physical Activity Monitoring,” *Sensors*, vol. 10, no. 8, pp. 7772–7788, 2010.

9. Appendix

9.1. Usability test questionnaire

Questionnaire as part of the usability test filled out by conducting researcher of the *NutriHEP* study.

1. Usability test of MoveIT

1.1. Interview

Please answer the following questions as detailed as possible. Feel free to just write single words, notes or drawings.

Program software

1. What is your first impression of the software?

Easy to understand.

Time efficient analysis of data.

Clear output.

2. What further functions or features do you miss?

Nothing, yet.

Diagrams

1. Which diagram do you prefer and why?

All diagrams serve a purpose. There are diagrams for just getting a good overview and diagrams to get information in more detail.

2. Is there any information which you see as unnecessary?

No.

3. Which information or graphics do you miss?

None.

Page **1** of **4**

User manual

1. What do you like about the user manual?

Most processes are described very clearly and are understandable in combination with using the software

Figures help to understand the processes where necessary

No unnecessary information

2. What do you dislike about the user manual?

Some points are missing --> therefore not always super clear how to proceed

Any further comments

Missing points in the manual were discussed with the author and will be resolved by the author.

1.2. Questionnaire

Please answer the following questions by ticking the appropriate scaling.

Program software

1. The menu looks neat.

I totally disagree ☹	I partially disagree	I can't decide 😐	I partially agree	I totally agree ☺
☐	☐	☐	☐	☑

2. The program explains itself.

I totally disagree ☹	I partially disagree	I can't decide 😐	I partially agree	I totally agree ☺
☐	☐	☐	☑	☐

3. The program will save me time in my work.

Very unlikely ☹	Fairly unlikely	Maybe 😐	Fairly likely	Very likely ☺
☐	☐	☐	☐	☑

Diagrams

1. The diagrams give a good overview of the data.

I totally disagree ☹	I partially disagree	I can't decide 😐	I partially agree	I totally agree ☺
☐	☐	☐	☐	☑

2. The diagrams are simple.

I totally disagree ☹	I partially disagree	I can't decide 😐	I partially agree	I totally agree ☺
☐	☐	☐	☐	☑

Page **3** of **4**

3. The diagrams explain themself.

I totally disagree ☹	I partially disagree	I can't decide 😐	I partially agree	I totally agree ☺
☐	☐	☐	☐	☑

4. I would use a diagram in my future work.

Very unlikely ☹	Fairly unlikely	Maybe 😐	Fairly likely	Very likely ☺
☐	☐	☐	☐	☑

5. I can quickly access the information I am looking for.

Very unlikely ☹	Fairly unlikely	Maybe 😐	Fairly likely	Very likely ☺
☐	☐	☐	☐	☑

User manual

1. The user manual is helpful.

Very unlikely ☹	Fairly unlikely	Maybe 😐	Fairly likely	Very likely ☺
☐	☐	☐	☐	☑

2. The user manual is understandable.

Very unlikely ☹	Fairly unlikely	Maybe 😐	Fairly likely	Very likely ☺
☐	☐	☐	☑	☐

3. Do you feel confident to work with MoveIT only with manual instructions?

Very insecure ☹	Fairly insecure	I can't decide 😐	Fairly confident	Very confident ☺
☐	☐	☐	☑	☐

9.2. User manual

The following pages contain the user manual for the developed software program.

DEUTSCHES ZENTRUM FÜR LUFT- UND RAUMFAHRT E.V. (DLR)

Movement Identification Tool (MoveIT)

User Manual

uwe.mittag@dlr.de

Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)
Institut für Luft- und Raumfahrtmedizin
Abteilung Weltraumphysiologie
Linder Höhe
51147 Köln
Deutschland

German Aerospace Center
Institute of Aerospace Medicine
Department of Space Physiology
Linder Höhe
51147 Cologne
Germany

Content

1. Introduction

The *Movement Identification Tool (*MoveIT*)* is an easy to use software program for analyzing daily physical activity of human beings. By using MoveIT, it is possible to adapt classifying systems to an individual person to view activity profiles or raw accelerometry data. Its use requires the application of USB accelerometry sensors by *Golf Coast Data Concepts*.
This manual shows how to install the MoveIT software, how to configure the accelerometry sensors, and how to start the analyzing procedures.

Minimum system requirements:

- *Golf Coast Data Concepts* accelerometers

Recommended system requirements:

- *Golf Coast Data Concepts* accelerometers
- *Mathworks MATLAB® R2012b*
    - *Neural Network Toolbox*
    - *Signal Processing Toolbox*

Formatting definitions:

Italics	Brand name
Bold	Folder or feature/button name
Underlined	Link to chapter in this manual
Underlined blue	Link to server folder or website

2. Installation

The *Movement Identification Tool* (MoveIT) software is based on and, therefore, only works with *Mathworks MATLAB®*. Before working with MoveIT you need to find out, if the computer you want to work on has a valid *MATLAB®* license installed. Additionally, you need the *Neural Network Toolbox* and the *Signal Processing Toolbox*. Any version since *MATLAB® R2012b* is recommended.
Ask your departmental IT administrator for further help.

2.1. Working on a computer with *MATLAB*® license

If you have access to a valid *MATLAB*® license and the essential toolboxes, you only need to install the MoveIT software.

Installation of MoveIT

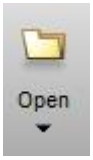
The software code is saved on the DLR internal server. It is saved in several *MATLAB*® compatible m-files.

1. Go on the DLR internal server to G:\WP\IT\MoveIT.
2. Copy the folder **Program Code** to your local hard drive.

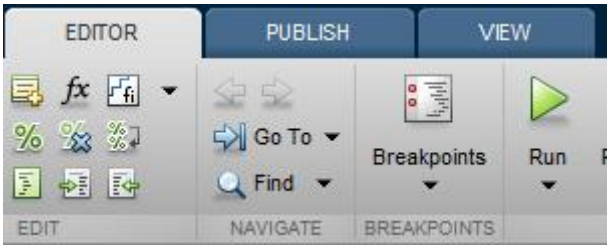
Starting MoveIT

To run the MoveIT program the code needs to be executed from *MATLAB*®.

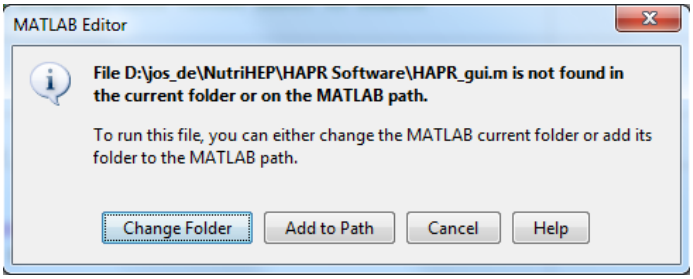
1. Start *MATLAB*®.
2. Press **Open**, go to the folder where the MoveIT software is saved and choose the file **MoveIT_main.m.**



3. On the tab **EDITOR** press on the green arrow saying **Run.**



4. If this is the first time you run the MoveIT software, the following window will pop up. Press **Add to Path** to continue.



5. The MoveIT software will start showing the home menu.
   (For more information, see Getting started with MoveIT on page xiii)

2.2. Working on a computer without *MATLAB*® license

If you do not have a valid *MATLAB*® license on your computer, you need to install the *MATLAB*® *Compiler Runtime* and the MoveIT software.

> ***Please note:*** Without a *MATLAB*® license it is only possible to work with pre-trained neural networks. Therefore, the feature to train a neural network is disabled.

Installation of MCR

MCR stands for *MATLAB*® *Compiler Runtime* and allows the execution of compiled *MATLAB*® applications on computers with no *MATLAB*® license.

1. Go on the DLR internal server to G:\WP\IT\MoveIT.
2. Copy the folder **Standalone Software** to your local hard disk drive.
3. Run the file **MCRInstaller.exe**.

> ***Please note:*** You will need administrator rights to run this file. Ask your departmental IT administrator for further help. The MCRInstaller.exe is also available online on http://www.mathworks.com/products/compiler/mcr.

Installation of MoveIT

The standalone application of the MoveIT software does not need further installation. Copying the executable file is enough.

1. Go on the DLR internal server to G:\WP\IT\MoveIT.
2. Copy the folder **MoveIT Software.exe** to your local hard disk drive.

Starting MoveIT

After successful installation of the MCR, the standalone file can be executed.

1. Run the file **MoveIT Software.exe**.
2. The MoveIT Software will start showing the home menu.
   (See Getting started with MoveIT on page xiii)

3. Configuration of accelerometers

Working with MoveIT requires certain configurations of the accelerometers. The following instructions refer to accelerometry sensors from *Gulf Coast Data Concepts (GCDC)*. Onboard software for configuration is included.

> **Please note:** For now, MoveIT only works with accelerometry sensors from *Gulf Coast Data Concepts*.

3.1. How to use the *XLR8R software*

The onboard configuration software *XLR8R* by *GCDC* is stored on the flash drive of every accelerometer. If you cannot find the software on the flash drive, download the latest version on http://www.gcdataconcepts.com/analysis.html. To run the standalone software, *JAVA 6.0* or later needs to be installed on your computer.
Ask your departmental IT administrator for further help.

Starting the XLR8R software

1. Plug in your accelerometer via USB
2. Go to the folder **xlr8r** and double click on **XLR8R.jar**.
3. The *XLR8R* software will start showing the home menu.

Please note: For further instructions how to use the XLR8R software refer to its help documentation stored on the accelerometer's flash drive or online on http://www.gcdataconcepts.com/XLR8R_r2_help.pdf.

3.2. Mandatory settings of accelerometers for the use with MoveIT

The following settings are mandatory for MoveIT working properly.

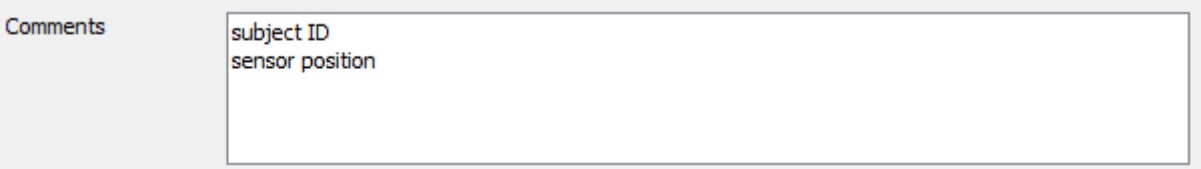
Setting time

1. Under **Utilities** open the tab **Set Device Time**.
2. Press the button **Write File**.
3. Immediately close the *XLR8R* window and safely unplug the accelerometer.
4. Start the accelerometer right away.
5. After a few seconds you can stop the acquisition.

   Why do I have to be so quick? – When the button **Write File** is pressed, the current computer time is saved in a txt-file on the accelerometer. On the next boot of the accelerometer, the time in the txt-file is loaded into the accelerometer's real time clock. This means that the time between pressing **Write File** and a completed reboot is equivalent to the absolute time shift of the accelerometer's internal clock.
6. The new generated data file in the **GCDC** folder of the accelerometer can be deleted.

Subject ID and sensor position

1. Open the *XLR8R* software (**XLR8R.jar**).
2. Under **Utilities** open the tab **Configuration File Editor**.
3. In the **Comments** section type in subject ID and sensor position in the following manner:



   It is important, that the subject ID appears in the first row and the sensor position in the second row.

   - For the NutriHEP study: write as sensor position “Hip” or “Tibia”.

4. Press **Save File**.
5. Close the *XLR8R* window.

4. Getting started with MoveIT

The MoveIT software offers several features for gaining information about a person’s activities. The following documentation will guide you through the preparing procedures.

4.1. How to set the default folder path

In MoveIT, files and folders often need to be chosen in a browser window. To avoid too much clicking along the same paths, you can define a default folder, where the browser window should start. There are two ways to change the default folder path:

A

1. Under **Default Folder Path** press **Browse…**.
2. Click through to you folder of choice.

B

1. Under **Default Folder Path** click in the text field
2. Type in your folder path of choice.

The program checks for the existence of the chosen folder. A default folder path has to be chosen to run any of the following functions.

> **Please note:** At start of MoveIT the default folder path is always set to G:\WP\STD\NutriHEP (NHP)\b Durchführung\Accelerometerdaten. A change of the path is only saved until the home screen window of MoveIT is closed.

4.2. How to import data

MoveIT contains a feature for retrieving data from an USB accelerometer to your hard drive or network drive.

1. Under **Preparation Phase** press **Import Data**.
2. Choose the USB drive of the accelerometer with the data you want to copy and press **Ordner auswählen**.
3. Subject ID, sensor position and date of acquisition is shown for verification
4. Choose whether you want to save it under the default folder path or a new path.
5. The data is saved with the following folder structure (if default folder path chosen):
    - Default folder path
        - Subject ID
            - Calendar week
                - Sensor position
6. The loading process can be canceled by closing the progress window.

5. Working with tibia and hip sensors

The development of MoveIT was conducted in consideration of the upcoming *NutriHEP* study at the *German Aerospace Center*. This study will monitor each subject via two accelerometers, placed at hip and tibia. This issue is faced in MoveIT in particular. The following documentation will guide you through several functions and their purposes for the *NutriHEP* study.

5.1. How to synchronize two accelerometers

For a proper classification with tibia and hip sensor the two accelerometers need to be synchronized, since the accuracy of the time setting (as in 3.2. Mandatory settings of accelerometers for the use with MoveIT on page xii) is insufficient. To synchronize two accelerometers a shared impact on both sensors is needed.

1. Turn both sensors ON
2. Slightly bump sensors against each other
3. Turn both sensors OFF
4. Connect first sensor to PC via USB
5. Open USB drive and go to **GCDC** folder
6. Copy last edited CSV-file
7. Open subject folder and create a new folder **Time Synchronization**
8. Paste CSV-file and rename it after the sensor ID (e.g. Acc0004)
9. Delete CSV-file on accelerometer.
10. Do steps 4 – 9 with second sensor
11. Open **MoveIT** software (See 2. Installation on page ix)
12. Press the button **Time Synchronization**
13. Browse to subject folder and choose CSV-files consecutively
14. If necessary, the name of the sensor can be changed
15. Both accelerometry data sets from the bumping impact in step 2 will be plotted in a new window
16. Click the peak on the graph of the blue line, so a little black square appears on that point, and press **Done**
17. Mark the corresponding peak on the green line and press **Done**
18. The program will synchronize the two graphs and will show the result in a new window.
19. If the result is unsatisfying, start over from step 12
20. If the result is satisfying, close the window
21. In the folder **Time Synchronization** a new TXT-file is generated containing information about the time synchronization
    - This file needs to be copied to the designated folder before classification (See 5.3. How to classify activities on page xvii)

5.2. How to train a neural network

At this point accelerometry data should be acquired from activities, which are later to be classified. These data files can be saved with the **Import** feature as described in 4.2. How to import data on page xiv.

> ***Please note:*** A *MATLAB*® license is needed to use the training feature. Without a *MATLAB*® license it is only possible to work with pre-trained neural networks. Therefore the feature is disabled.

To accelerate the progress the subject can complete a default training course as it is described in Attachments on page xx. This course contains several activities in a specific order.

To train a new neural network, follow the instructions below:

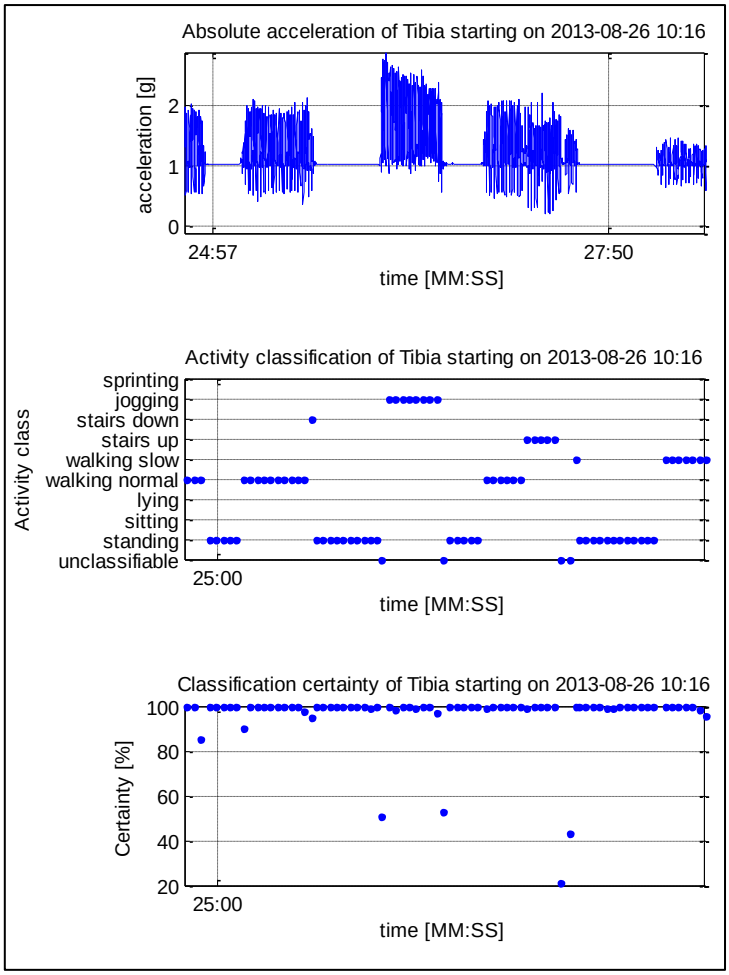
1. Import the training data using the **Import** feature of MoveIT
2. Change the folder name of the calendar week into **Training**
3. Press the button **Training** on the home screen of MoveIT
4. Browse to subject folder and choose the **Training** folder
    - Chosen folder must contain folders named **Hip** and **Tibia**
5. The opening window is for identifying activities in tibia data
6. If the 7.1. Default training course on page xx was completed,
    - the pink-green-colored template should approximately fit to the accelerometry data
    - static activities are coded green, and dynamic activities are surrounded by pink lines
    - the vertical lines can be dragged in the right position, so the activity is framed
    - the activity identification with the template can be saved with **Save template**
7. If a custom training course was completed,
    - the activity template can be disabled
    - activities can be framed with the feature **Select activity data**
    - the square framing must include the red dots, which are indicating the dynamic nature of the accelerometry data
    - blue lines of acceleration do not need to be framed
    - press **Yes**, if selection is done
    - selections cannot be adjusted afterwards
    - wrong selections have to be reframed by pressing **Redo Selection**
    - Undo of selection is not possible and requires a start over from step 3
    - new activities can be added with **ADD NEW ACTIVITY** and must be named and assigned to static or dynamic activities
    - acceleration data must not be selected twice
8. When finished, press **DONE**
9. Do steps 6 – 8 with hip data
10. The new neural network is saved as a MAT-file in the **Training** folder which was chosen in step 4.

5.3. How to classify activities

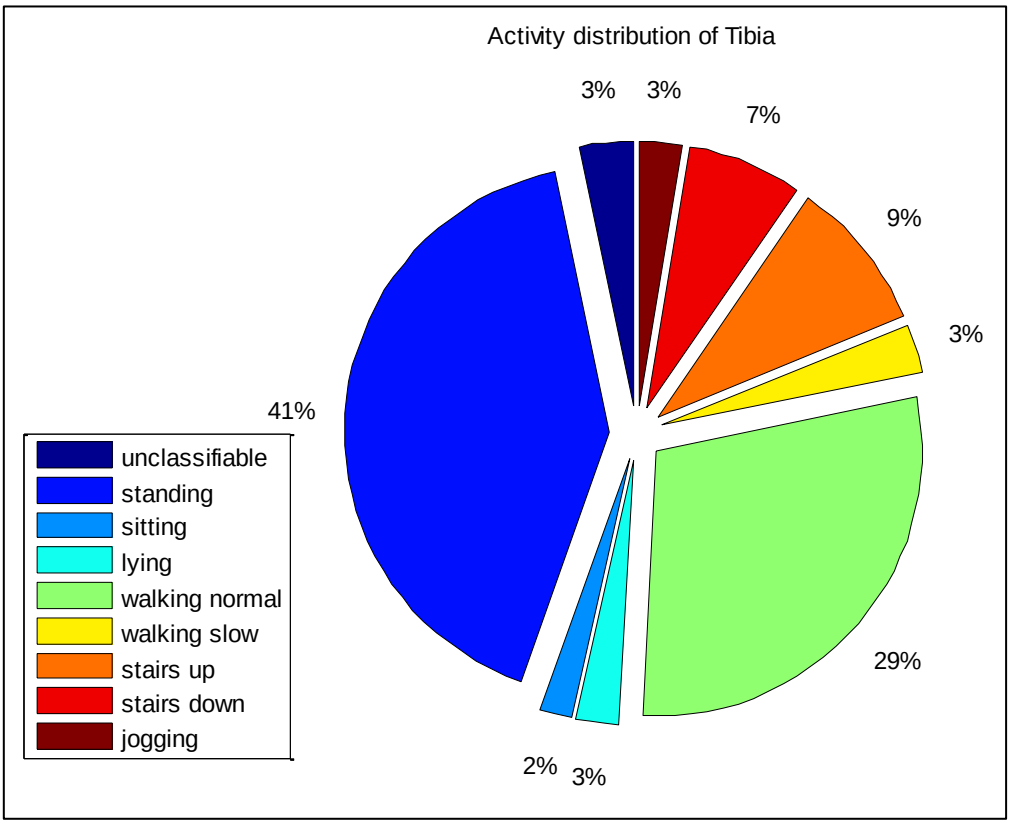
Follow the instructions to apply the classification procedure on accelerometry data:

1. Press the button **Classification**
2. With the **Browse…** buttons choose
   - a MAT-file as neural network
     (e.g. NeuralNetworks_SubjectID_date.mat)
   - a folder containing **Hip** and **Tibia** folder with data for classification
3. Define where the asked sensor is placed
4. If the error message “No time synchronization file found!” appears, follow step 20 in 5.1. How to synchronize two accelerometers on page xv and start over
5. Pop-up windows will show the classification results
   - Window with tibia results will appear exactly behind window with hip results
6. The window **Visibility of graphical representations** offers to choose which pop-up windows are shown on the screen.
7. To end the classification progress press **Close All**
8. The graphical representations are saved as FIG-files in a new folder

Graphical representations



On the left of the screen, the results are represented in a time dependent diagram. The upper curve shows the absolute acceleration of the measurement in g-force. The graph in the middle has a nominal scale with dots showing successful classifications. The lower graph contains the corresponding certainty levels of the classifications.



In the middle of the screen, the activity distribution is represented in a pie chart. Percentages refer to the total time of data acquired.

	Days : Hours : Minutes : Seconds		
elapsed time	00 : 00 : 14 : 59.565		
	Avg. certainty of Classification [%]		
dynamic activities	99.7922		
static activities	95.7240		
all activities	97.5903		
	No. of recognitions	Activity distribution [%]	estimated Time [dd:hh:mm:ss]
unclassifiable	10	3.3003	00 : 00 : 00 : 29
standing	125	41.2541	00 : 00 : 06 : 11
sitting	6	1.9802	00 : 00 : 00 : 17
lying	8	2.6403	00 : 00 : 00 : 23
walking normal	88	29.0429	00 : 00 : 04 : 21
walking slow	9	2.9703	00 : 00 : 00 : 26
stairs up	28	9.2409	00 : 00 : 01 : 23
stairs down	21	6.9307	00 : 00 : 01 : 02
jogging	8	2.6403	00 : 00 : 00 : 23
sprinting	0	0	00 : 00 : 00 : 00
SUM	303	100	00 : 00 : 14 : 59

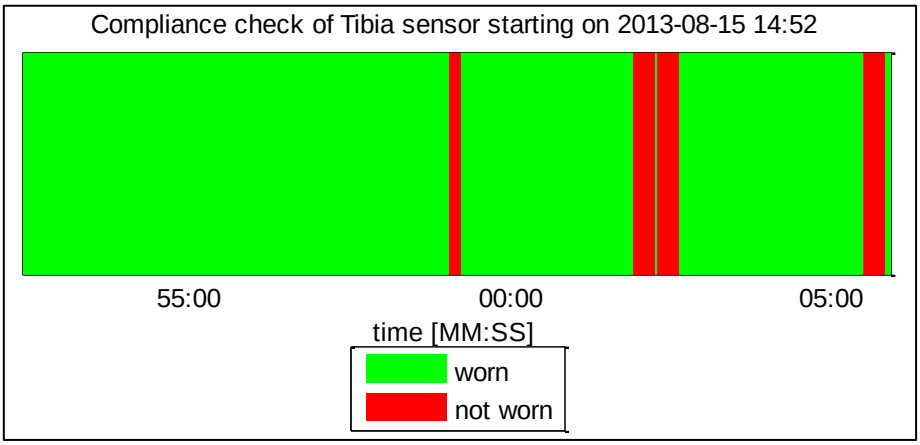
On the right of the screen, the distribution is shown again in absolute numbers organized in a table. It also contains the total time and the average certainty level of classification

5.4. How to check for subject's compliance

MoveIT provides the feature to check for the subject's compliance by verifying if hip and tibia sensor were worn simultaneously.

1. Press the button **Compliance**
2. Browse to subject folder and choose folder containing **Hip**, **Tibia**, and **Classification** folder
3. Pop-up window shows the results of the compliance check
4. Its graphical representation is saved as FIG-file in the folder chosen in step 2

Graphical representation



The compliance is represented in a colored timeline, where green and red stand for timeframes with worn and not worn tibia sensor.

6. Working with a single sensor

MoveIT is also designed for working with a single accelerometry sensor for usage outside of the *NutriHEP* study. The training und classification procedures with a single sensor correspond to the instructions above in 5.2. How to train a neural network on page xvi and 5.3. How to classify activities on page xvii. The training feature just needs a naming of the sensor's position beforehand. This naming is important since the software looks for the data in a folder of the same name. The steps of time synchronization and compliance check are not applicable.

7. Attachments

7.1. Default training course

Location	No.	Starting time	Activity	Description	Duration
Base-ment	**1**	0:00	*Standing*	- face walking direction - turn accelerometers ON - stand still and wait for climbing stairs	30 s
Staircase	**2**	0:30	*Climbing ascending stairs*	- walk 4 floors upstairs - keep uniform pace - at 3rd floor stop and stand still	~ 90 s
			Interstation activity	- stand still for 15 seconds - turn around - stand still and wait for 2:30	~ 30 s
	3	2:30	*Climbing descending stairs*	- walk 4 floors downstairs - keep uniform pace - no skipping or jumping - at end of stairs stop and stand still	~ 90 s
Base-ment			Interstation activity	- stand still for 15 seconds - walk to end of tunnel and turn around - stand still and wait for 5:00	~ 60 s
	4	5:00	*Walking with normal pace*	- walk on own pace - go around corner until black door - stop and stand still	~ 90 s
			Interstation activity	- stand still for 15 seconds - turn around - stand still and wait for 7:00	~ 30 s
	4	7:00	*Walking with normal pace*	- walk back on own pace to end of tunnel - stop and stand still	~ 90 s
			Interstation activity	- stand still for 15 seconds - walk to running start - stand still and wait for 9:30	~ 60 s
	5	9:30	*Running as jogging*	- jog to end of corridor - go around corner until black door - stop and stand still	~ 30 s
			Interstation activity	- stand still for 15 seconds - walk to ground floor - stand still and wait for 11:30	~ 90 s
Ground floor	**6**	11:30	*Walking with slow pace*	- walk slowly across corridor - stop and stand still at physiology lab door	~ 30 s
			Interstation activity	- stand still for 15 seconds - walk into physiology lab - walk to chair and stand back	~ 60 s
Physiolo-gy Lab	**7**	13:00	*Sitting*	- sit down on chair - both feet touch ground	30 s
			Interstation activity	- get up from chair - walk to couch and stand back	~ 30 s
	8	14:00	*Lying*	- lie down on couch - lie on back - do not cross legs	30 s
			Interstation activity	- get up from couch - walk back to chair - turn accelerometers OFF	~ 30 s