Machine Learning for Active Gait Support with a Powered Ankle Prosthesis

Maschinelles Lernen zur gesteuerten Gangunterstützung durch eine aktive Fußgelenkprothese

Master-Thesis von Simon Barnikol

Tag der Einreichung:

1. Gutachten: Prof. Dr. Jan Peters

2. Gutachten: Mahdy Eslamy / Prof. Dr. André Seyfarth

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Informatik
Intelligent Autonomous Systems

Machine Learning for Active Gait Support with a Powered Ankle Prosthesis
Maschinelles Lernen zur gesteuerten Gangunterstützung durch eine aktive Fußgelenkprothese

Vorgelegte Master-Thesis von Simon Barnikol

1. Gutachten: Prof. Dr. Jan Peters
2. Gutachten: Mahdy Eslamy / Prof. Dr. André Seyfarth

Tag der Einreichung:

Erklärung zur Master-Thesis

Hiermit versichere ich, die vorliegende Master-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 31. Oktober 2014

_____

(Simon Barnikol)

Abstract

In recent years active prosthetic devices emerged as research area. In comparison to passive devices, active ones mimic the motion of our intact limbs more perfectly. Hence, they overcome several drawbacks a user of a passive devices is confronted with. Examples for benefits caused by switching from a passive to an active prosthesis are a more normal gait and reduced metabolic costs of transport. Even though active devices can improve amputee's every day life, there are still challenges to be solved. Prosthesis control is besides design one of those challenges. The controller needs to decide when and how to adapt active support. Regarding, for instance, the push off phase, the active push off must be aligned with the timing and strength intended by the user. This work introduces recent supervised machine learning methods such as Gaussian process regression and support vector machines for control of active prosthetic devices. The machine learning methods are used to form a supervisory controller that infers the user's intent. As input to the supervisory controller, we use the data obtained by an inertial measurement unit mounted at the shank of the considered active ankle prosthesis. The output or rather the users intent is given by gait, speed and gait percent predictions. If the intent is known, the desired nut position can be determined by a lookup. To enforce the desired trajectory a slave controller, here a PD controller, is applied.

The supervisory controller is designed, implemented and tested based on walking and running data recorded on a treadmill. At first, noise free motion capturing data is used to demonstrate the applicability of supervised machine learning methods in context of active ankle control. Afterwards, sensor data obtained with the prosthesis's intertial-measuremt unit is used to proof real-world applicability of the introduced supervisory controller. The obtained supervisory controller is fast and moreover accurate. For gait percent prediction the error is bounded to $\pm 5\%$. In case of speed and gait prediction, accuracies close to 100% and 95% are achieved, respectively.

Contents

1 Introduction

Regarding Germany, there are no registers for amputees and no official statistics about amputation frequency. However, it is possible to estimate the frequency of lower limb amputations using clinical and health insurance data. In 2001, 45,000 amputations were performed, in 2002 and 2008 there were 55,000 and 60,000, respectively [1, 2]. The leading causes of amputation are vascular disease and diabetes. For instance, approximately 70 to 80% of all amputations performed in Germany in 2001 are due to diabetes [3]. In the future the prevalence of diabetes is growing world wide [4], so it is likely that the number of amputations will grow as well.

Most of the transtibial amputees use passive and quasi-passive ankle prosthesis. Both types only store the energy produced at the beginning of a stance and release it in its end. An intact human ankle, in contrast, is able to produce net positive work during stance. Consequently, passive devices mimic human gait imperfectly and impose energy deficiencies on their users. This limitations of passive devices, in turn, lead to locomotion problems like slower self-selected walking speed, asymmetrical gait and a higher metabolic energy consumption [5]. Moreover, people with an unilateral transtibial amputation are predisposed to musculoskeletal injuries, since they try to overcome the energy deficiencies with their unaffected leg, resulting in greater forces on the unaffected side [6].

To improve amputee ambulation, active ankle, also knowns as powered ankle, prosthesis were introduced. Such active devices can produce net positive work, and hence can mimic the human ankle more accurately. The improvements due to active devices were demonstrated in several case studies. For example, Au et al. have shown that an active ankle prosthesis leads to a more normal gait and reduces the metabolic costs of transport [7]. Nevertheless, the broad commercialization of active ankle prostheses is hindered by two challenges. The first challenge was addressed in the last few years and is about shrinking the prosthesis to human size and weight. This can be done by decreasing peak power and energy consumption of the active ankle prosthesis, and thereby also the motor, battery and prosthesis size [8]. Inspired by this approach, several prosthesis designs, e.g., SEA, SEDA (series elastic-damper actuator) and PEDA (parallel elastic-damper actuator), emerged. All approaches are abbreviated with the mechanical concepts they are based on. SEA, for instance, stands for series-elastic actuator and is implemented as mechanical transmission in case of a motor in series with a spring. Recently it was shown that SEA is the best compromise if different human gaits, like normal level, upstairs and downstairs walking, are considered [9]. The second challenge is the control of powered ankle prostheses. Only with appropriate control mechanism, it is possible to fully leverage the potential of active devices. For instance, an powered push off is only worthwhile, if it is performed at the right time. Until now, several control strategies were proposed but no strategy established as standard. For instance, Varol et al. use finite-state impedance controllers to execute a desired trajectory. To detect the desired trajectory or rather the user's intent they designed an overlaying supervisory intent recognizer [10]. In contrast, Holgate et al. use the tibia angle and velocity to calculate gait percent. Given gait percent and stride length in addition, they can determine a unique nut position which is applied to the prosthesis [11]. A more detailed literature review is given in Section 1.3.

This work introduces a new control strategy for a powered ankle prosthesis. The control strategy is based on an overlaying controller which uses supervised machine learning to infer the user's intent. Given the intent, the prosthesis achieves best possible user support by adapting to it. A more detailed description of controller design and objective is given in Section 1.1. The two remaining sections introduce the structure of this thesis and give an overview of related work.

1.1 Objective

The goal of this thesis is the design and implementation of a new control strategy for a powered ankle prosthesis. As depicted in Figure 1.1, the approach is constituted by a supervisory controller, a lookup and a slave controller. The supervisory controller, or rather master controller, determines the user's intent based on online sensor data provided by the prosthesis. When the intent is known, a simple look up gives the corresponding desired trajectory. The slave controller, in turn, is used to enforce the desired trajectory to the prosthesis. For the considered prosthesis, the slave controller is a PD controller and the desired trajectory is given by the nut pattern applied to the prosthesis. Note that the lookup procedure and the slave controller are not investigated further. For the lookup method we refer to a procedure given in [11] and for the design of the slave controller standard methods of control theory are applied.

The focus of this thesis is on introducing a supervisory controller which uses supervised machine learning methods to infer the user's intent. As input to the controller, the prosthesis provides data of an inertial measurement unit mounted at its shank. Given this data, the supervisory controller performs three predictive tasks such as gait, speed and gait percent prediction. The outputs of all tasks in conjunction are synonymous with the user's intent. From a machine learning point of view, the functionality of the overlaying controller can be described as follows: At first, classification is used to recognize the intended gait. Afterwards, the corresponding speed is determined with the use of classification or regression. Classification is used if the speeds are discrete and their amount is countable. Otherwise regression is used. For gait percent estimation regression is used as well.

At first, the supervisory controller shall be designed and implemented based on walking and running data provided by a motion capturing system. Afterwards, the resulting controller must be evaluated for real-world applicability as well. Therefore, motion data recorded by

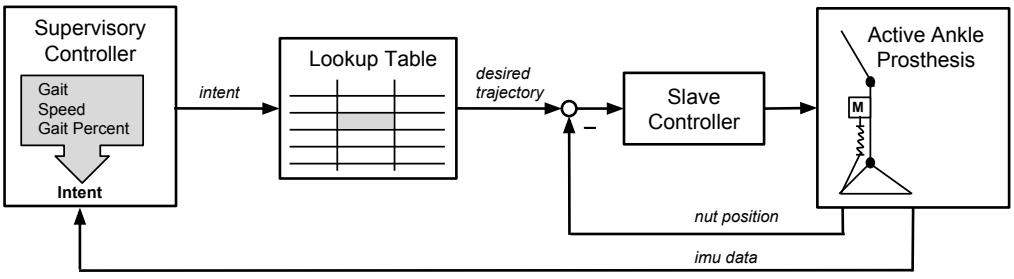


Figure 1.1.: Overall control architecture. The supervisory controller is based on machine learning methods and infers intended gait, speed and gait percent. Given the user's intention, a lookup table and a slave controller are used to adapt the prosthesis's trajectory to it.

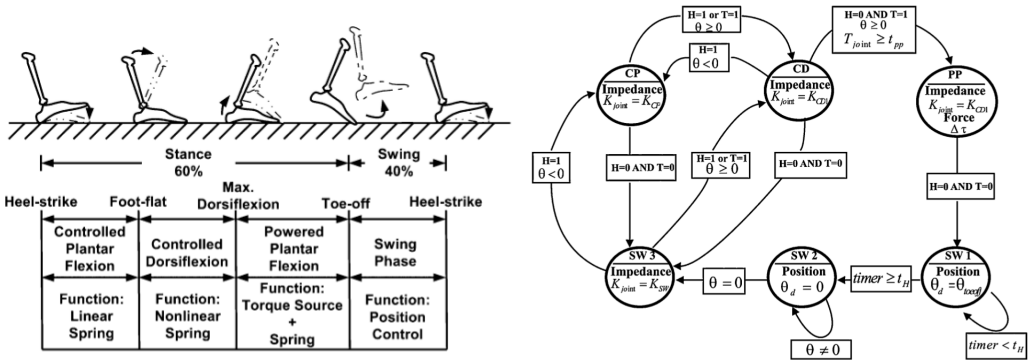
the prosthesis's inertial measurement unit is used. The implemented controller should meet the basic requirements such as accuracy and speed and should require as less input signals as possible. To optimize the controller with respect to these requirements, different setups are evaluated. Furthermore, different sensor signal combinations and different frame sizes, e.g., the data of the last 5 or 10 time steps, can be used. Most likely, accuracy increases when using more sensor data and a higher frame size, but meanwhile speed may decrease.

1.2 Outline

Chapter 2 introduces the machine learning background required for the presented control approach. The subsequent chapter analyses the applicability of machine learning as control mechanism. Therefore, motion data recorded by a motion capturing system is used. Such kind of data is very accurate and, in contrast to sensor data, noise free. As result of the applicability analysis, data that serves as input for gait prediction is identified and a control mechanism is proposed. In Chapter 4, real sensor data is used to proof that the presented concepts also work for wearable prostheses. Moreover, a transfer function is introduced which dramatically improves speed detection. The thesis concludes with Chapter 5, where the machine learning based control mechanism for active gait support with a powered ankle prosthesis is summarized and discussed. In addition an outlook an possible future work is presented.

1.3 Related Work

Before designing an own control approach, already published approaches for controlling active prostheses are reviewed. One of the first approaches is echo control, where the motion of the sound side leg is recorded and reproduced with the active prosthesis [12]. Because of the playback, the user is always forced to repeat the action of the unaffected leg which can be uncomfortable in some situations. Moreover, convenience is reduced by the additional sensors required at the unaffected leg. Other approaches that require to mount additional sensors are electromyography (EMG) or even nervous system based approaches. Electromyography based approaches mount surface EMG sensors to extract and use muscle activation signals for control. In this context, the non-stationary nature of EMG signals imposes some challenges for control. Examples for EMG based control approaches are given in [13, 14]. Nervous system control is the invasive counterpart of EMG based control and gives more accurate signals. Here, electrodes are either implanted at the central nervous system or at the brain [15, 16]. All approaches given above require sensors or electrodes in addition to the prosthesis. Consequently, a remaining approach is about operating and controlling the prosthesis with just the sensors attached to the device. Such approaches are considered here in more detail. Regarding transfemoral or transtibial amputees, the groups at MIT, Vanderbilt University and Arizona State University were especially concerned about the design, implementation and control of such prosthetic devices. In the following, the control approaches of those groups are introduced. Afterwards, we present some work unrelated to prosthesis control but with focus on similar problems as we are facing here.



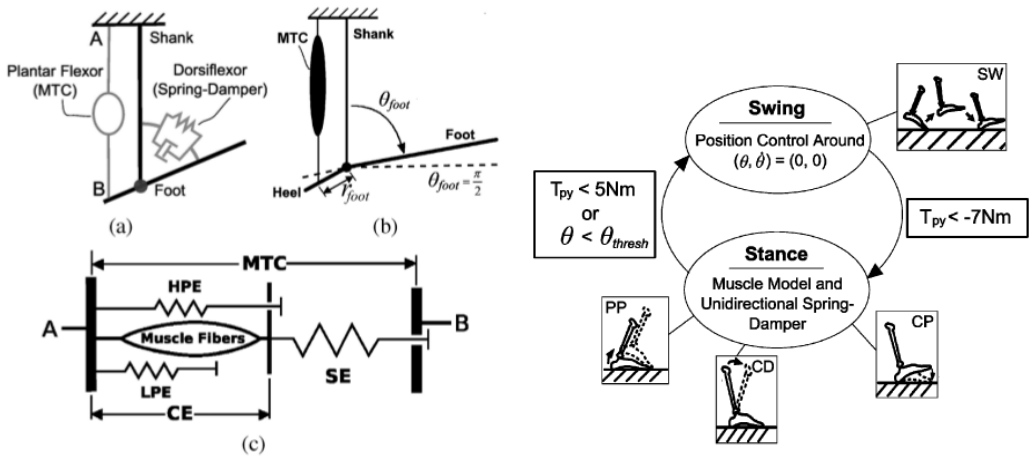
(a) State-based prosthesis design. The figure is taken from [17]

(b) State machine for level-ground walking. The figure is taken from [17]

Figure 1.2.: Panel (a) shows the state-based prosthesis design proposed in [17]. Each state is associated with a different mechanical function and control approach. (b) illustrates the state machine for level-ground walking in detail. Circles denote states and rectangle state transition conditions. Depending on the state, position or impedance control is applied.

MIT Media Lab

First, the research of Hugh Herr's biomechatronics group at MIT Media Lab is considered. In [17] the authors describe the implementation of a powered ankle–foot prosthesis and the corresponding control mechanism. The prosthesis is designed and controlled to mimic the human motion of an intact limb more perfectly than a passive device. By doing so, amputees can achieve a more normal gait and reduce their metabolic cost of transport. Note that the authors of [17] consider only level-ground walking. Prosthesis design and control is driven by a state based approach. Therefore, the gait cycle is divided into different phases or rather states. In context of design, each state is associated with a different mechanical function. For instance, controlled plantarflexion is associated with a linear spring. For more details about the definition of states and how they are used in prosthesis's design see Figure 1.2a. Also for control, the state-based approach is chosen. For each of the states a specific controller is designed and fine-tuned. Common control approaches are position control, e.g., for the prosthesis's nut position, and impedance control. The authors of [17] rely on both, depending on the state. The control scheme consists of six different states. The corresponding state machine is given in Figure 1.2b and explained in the following. For the stance phase, the three states defined in Figure 1.2a are used. Controlled plantraflexion begins at heel-strike and lasts until midstance. Afterwards the second state, controlled dorsiflexion, continues until the ankle torque reaches some threshold. The last state of the stance phase is powered plantarflexion. Each state is controlled to fulfill a special task [17]. Controlled plantaflexion is about absorbing the power of heel strike, controlled dorsiflexion and powered plantaflexion are about rotating the body and powered push off, respectively. In contrast to the prosthesis's design, the swing phase is divided in three states as well. The first states begins at toe-off. After some time period is exceeded, the second state starts and lasts until ankle angle reaches zero. The third state is active until the next heel strike occurs. Regarding the swing phase, the control task is mainly about positing the prosthesis for the next heel strike. To switch between the states, sensors inputs are required. As inputs the authors use the ankle angle and heel and toe contact sensors [17]. The angle ankle is denoted with θ and the heel and toe contact sensors are abbreviated with H and T. If H or T equals 1 a contact is detected,



(a) Neuromuscular model used for control. The figure is taken from [18]

(b) Finite-state machine used for control. The figure is taken from [18]

Figure 1.3.: (a) depicts the components of the neuromuscular model. The top left figure displays the neuromuscular model and the top right figure its geometry. The figure at the bottom shows the Hill-type muscle used by the model. (b) shows the state machine used for neuromuscular model based control

otherwise a contact is not given. All in all, the prosthesis performs good, since it is able to mimic the motion of an intact ankle more perfectly than a passive device [17]. This motion, however, changes with walking speed. To mimic also different speeds, the controller needs to be fine tuned to those speeds. The fine-tuning is possible, but the presented approach is not able to infer the speed intended by the user.

The same group is involved in another approach aiming at a more adaptive controller [18]. As testbed an active ankle-foot prosthesis of iWAlk LLC, which is a successor of the MIT Media Lab prosthesis, is used. The controller is based on a neuromuscular model of the human-ankle foot complex and directly generates the ankle torque required for control. Figure 1.3a depicts the neuromuscular model (a), its geometry (b) and the used Hill-type muscle model (c). The human ankle is modeled by a hinge joint controlled by two virtual actuators. For the first actuator an unidirectional plantarflexor, given by a Hill-type muscle model, is chosen. From a controller point of view, the Hill-type muscle includes a positive force feedback reflex scheme which is used to incorporate adaptivity. The second virtual actuator is a dorsiflexor that is mostly active during early stance phase. Furthermore, the dorsiflexor is divided into two states. One state is covered with a reactional proportional-derivative position controller, whereas the second state uses a unidirectional virtual rotatory spring-damper. As for the previously presented control approach, the state depends on the gait phase [18]. To generate the control signal, both virtual actuators produce different torques depending on the state. The net torque at the ankle joint constitutes the final control signal. Considering the used prosthesis, the torque is produced by a parallel spring and a motorized drive train. An angle sensor gives the torque produced by the spring. The net torque is achieved, since the remaining amount is given by the motor controller. For maintaining the controller's state, the state machine shown in Figure 1.3b is applied. The state machine distinguishes between swing phase, and stance phase. In case of stance phase controller plantarflexion (CP), controlled dorsiflexion (CD) and powered plantarflexion (PP) are considered. To identify state transitions, the ankle torque T_{py} and ankle angle θ are used as input signals. The neuromus-

cular model has free parameters that must be fitted to give good performance. As remainder, good performance is given when the intact ankle behavior can be mimicked closely. Here, the parameters of the model are optimized for level-ground walking at $1.0m/s$. Therefore, corresponding motion capturing data was recorded with a healthy, male subject of 81.9 kg weight. For optimization, the authors used a cost function given by the squared error between biologic and controller-based trajectory. A genetic algorithm selects the initial optimization parameters and direct search gives the final results. The resulting controller was tested with an transtibial amputee of 75kg weight. When walking at $1.0m/s$, the amputee's motion was close to the recorded intact motion [18]. Even though the controller does not explicitly sense its environment, it was also able to adapt to ramp ascent and descent scenarios. For ramp ascent, the produced net work increases and for ramp descent it decreases. This can be observed for intact humans as well. The pure force feedback reflex scheme, however, is not sufficient to adapt to different speeds. That is why the model is extended with length and velocity feed-back terms [19]. The resulting neuromuscular model is only a coarse approximation of the human model, but can adapt to different speeds.

CIM at Vanderbilt University

The center of intelligent mechatronics at Vanderbilt University is also developing active prosthetic devices. The group, led by Michael Goldfarb, implemented an electrically powered transfemoral prosthesis which is controlled by finite-state impedance control. How the prosthesis adapts to the user's intent is described in [20]. The described mechanism is constituted by two controllers, namely a supervisory and an intra-modal controller. Figure 1.4 display the control structure. The supervisory controller analyzes the sensor signals to detect the user's intent. When the intent is known, a corresponding finite-state impedance controller is used to support the user. So, there are several intra-modal controllers, fine-tuned on specific scenarios and the supervisory controller is responsible for picking the right one. The user's intent is given by gait, cadence and slope estimators. In the referred paper [20], the gait mode estimator is described. Gait mode estimation discriminates between standing and walking. First, the intra-modal controllers are fined tuned for the test subject. The test subject is a healthy male person that uses the prosthesis with an able-bodied adapter. For the intra-modal controllers, scenario such as standing and slow, intermediate and fast walking are considered. In a next

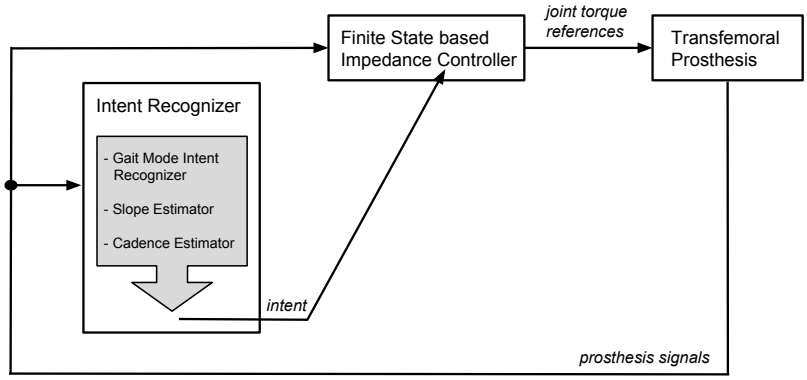
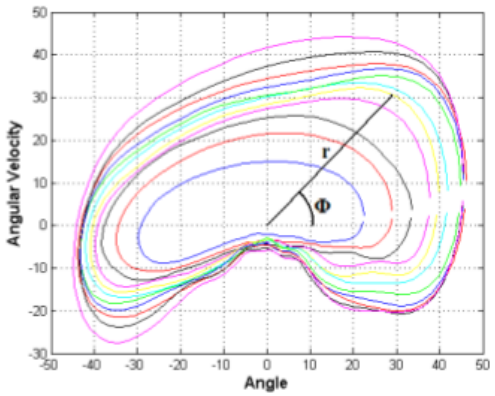


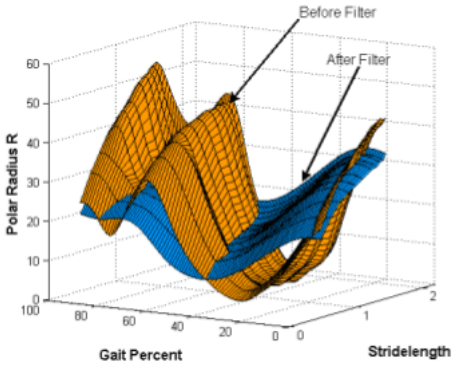
Figure 1.4.: Overall control structure for a powered transfemoral prosthesis developed at Vanderbilt University. Note that the figure is adapted from [20]. The depicted supervisory controller detects gait mode, slope and cadence and chooses a corresponding intra-modal controller for prosthesis control.

step, the authors generate a database for the considered scenarios. The database covers also scenarios where the intra-modal controller and the user's intent do not match. All recordings were performed at 1000 Hz and are constituted by prosthesis's signals like knee and angle joint positions and velocities, socket sagittal plane moments, and heel and ball forces. Since the gait estimator is optimized with respect to different properties, four trials are recorded for each database element. The properties result from the different steps required for gait estimation. In a first step, a specific frame length is chosen. If, for instance, the frame length is set to 50, the last 50 measurements of each sensor are used as input for gait estimation. In a second step, features for gait estimation are calculated. The authors chose the mean and standard deviation of each input signal's frame as features, since both are computationally inexpensive. Also the next step, namely a dimensionality reduction, is preformed to decrease computation time [20]. The results of dimensionality reduction are used as input for the estimator itself. Estimations are performed with Gaussian mixture models (GMMs). Therefore, a Gaussian mixture model with k components is learned for each gait mode. The expectation maximization algorithm is applied to learn the Gaussian mixture models. For each gait mode, the algorithm is initialized with the results obtained from k-means clustering performed on the gait mode's data. During operation, new inputs are classified by evaluating the mixture models for each gait and choosing the most probable one as prediction. The authors observe, that sometimes predictions are temporarily wrong or chatter between walking and running. That is why, they introduce a voting scheme as last processing step. The voting scheme stores the last l prediction results and only gives new classification results if at least 90% of those predictions are in accordance. Consequently, the properties considered during controller optimization are frame length, dimensionality reduction, number of mixture components and voting scheme length. As frame length 50, 100, 200 and 400 samples are considered. For dimensionality reduction either principal component analysis or linear discriminant analysis are used. As number of Gaussian mixture models and voting scheme length $k \in \{2, \dots, 8\}$ and 2 to 100 are examined, respectively. The authors of [20] report that the best performing model is given by a GMM with 7 mixtures, a frame size of 100 samples, a three dimensional principal component analysis and a voting length of 38. Note that voting scheme length and frame length together define the total classification delay. To test the best performing supervisory controller, it was implemented and tested on the prosthesis. In doing so, the same subject performed ten trials of level-ground walking on a treadmill. During all trials the subject walked at a constant speed. At random time steps the treadmill was stopped so that the subject changed its gait mode to standing. Afterwards the treadmill was started again. This procedure was repeated to observer several gait mode switches. All in all, the authors observer that no wrong mode switches were performed by the controller. In [10] the authors extend the presented gait mode classification also to sitting. Here some errors were observed during evaluation. The errors, however, had no impact to the user of the prosthesis, since they occurred only for very similar activities [10].

The slope estimators for standing and walking are described in [21, 22]. For standing, an inertial measurement unit is attached to the prosthetic foot. Based on their sensor measurements, the ground slope is, for an range of 15%, estimated with an error of $\pm 1\%$ [21]. For walking also an inertial measurement unit is required. In addition, heel and ball load sensors are used to detect ground contact. The ground contact is a reference for determining acceleration magnitude which is given by gravity. By observing the change of acceleration along the axis orthogonal to the foot, the slope is estimated [22].



(a) Tibia angle-velocity cyclogram. Figure taken from [11]



(b) Surface plot for polar distance, polar angle and gait percent. Figure taken from [11]

Figure 1.5.: (a) Tibia-angle plotted against angular velocity multiplied by a scaling factor, and (b) surface plot for polar distance, polar angle and gait percent. In contrast to the orange surface plot, the blue one is invertible. So it can be used to determine stride length given polar angle and gait percent. The blue surface plot results from the orange one by applying a first order filter to the polar radius.

Department of Engineering at Arizona State University

Another active prosthesis, known as SPARKy, is developed by Thomas Sugar's group at Arizona State University's Polytechnic Campus. This group proposed also different control methods. A first method is known as robust control and is applied for a powered ankle-foot orthosis [23]. The method is based on velocity and stiffness control. In more detail, the stance phase is divided in five different zones. Each zone is recognized by an unique event, e.g., zone 1 begins at heel strike and zone 2 starts when ankle angular velocity crosses zero. Depending on the zone, either velocity or stiffness control is applied. By changing the stiffness or velocity parameters of a zone, different output profiles are achieved. In future work, the authors want to investigate if they can achieve different activities than walking, e.g., stair ascent, by just varying these parameters [11].

A second control approach, known as tibia based control, is about determining gait percent and stride length [11]. If both are know, the motor pattern for level-ground walking is given by a simple look up. The approach is based on the cyclogram given in Figure 1.5a, where tibia angle is plotted against tibia angular velocity multiplied by a scaling factor. Regarding Figure 1.5a, different line colors denote different speeds and the larger the curve the larger stride length. Instead of using Cartesian coordinates, the authors of [11] switch to a polar coordinate representation. This representation reveals two useful analytical relationships. First, polar angle is directly related to gait percent. Consequently, the authors use the cyclogram to fit a function for gait percent computation. This function takes polar angle, determined from tibia angle and velocity, as input and returns gait percent. Second, the polar radius increases with longer stride length. The polar radius distance, however, depends on polar angle, as well. For instance, for a polar angle of 90 degree all distances are relatively small compared to the distances for a of polar angle of 270 degree. The authors condensed the relationship of polar distance, polar angle and gait percent into a surface plot [11]. In Figure 1.5b, the resulting surface is colored orange. In contrast to the blue surface, the orange one is not invertible. The blue surface is achieved by applying a first order filter to the polar radius. Since the blue surface is invertible, a function that consumes polar angle and gait percent

and yields stride length can be found. When gait percent and stride length are known, the prosthesis's control signal can be determined with a look-up table, also defined by Holgate et al. [11]. In this context, the control signal is the desired nut position. To compute stride length and gait percent, the presented approach requires actual measurements for tibia angle and velocity. The used prosthesis, however, is only equipped with an angular rate sensor. So, tibia velocity is known, but tibia angle is not. The authors compute tibia angle with the use of a pseudo integrator. The pseudo integrator is a transfer function that gives results similar in shape but not identic to real tibia angle. Note that similarities in shape are enough to apply the presented approach. Moreover, the authors identify several benefits resulting from the transfer function [11]. First, the method is easier to apply than a Kalman filter or strap down integration. Second, the transfer function is stable so that there is no need for resetting integration bounds. Furthermore, the tibia angle is normally unique for each subject. A third benefit given by the transfer function is that its output incorporates deindividualization to some degree. The presented control approach was implemented and tested on the SPARKy robot attached to an amputee. During testing, the amputee walked speeds ranging from slow to fast. The results for stride length estimation are with in an error of 10% and the gait percent error is bounded to $\pm 5\%$.

Speed Prediction Based on Inertial Measurement Sensors

Concluding, we present a research topic unrelated to prosthesis control, but also concerned with a task we are facing here. This research is about inertial sensor-based methods for speed estimation. Since the active ankle prosthesis is equipped with an inertial measurement unit as well, work published in this field might scale for prosthesis control. In [24] a systematic review of speed prediction based on inertial sensors is given. The review categorizes the existing approaches in classes such as abstract models for speed prediction, human models and direct integration. In case of abstract models, artificial neural networks (ANN) were used for speed estimation. Depending on the overall task, e.g. if incline waling, running, or other extensions are considered, different numbers of ANNs and different layer configurations per ANN are used. Moreover, different sensor locations are possible. In [25] the sensor is attached to the chest and various speeds in between 4.7–17.14km/h are considered. The overall root mean squared error (RMSE) is reported to be 0.54km/h. In contrast to those black box approaches, human gait models are based on analytical models of the human leg. For instances in [26], the human leg is given by just one joint representing a simple pendelum. Here, the sensor is required to compute the leg angle. Given the angle and the leg model, the stride length is computed analytically. So, the speed prediction results depend on model accuracy. A more sophisticated model is given in [27], where the human leg is modeled by two joints. The direct integration approach is mainly used for personal navigation, e.g. to identify the location and speed of a user within a room. In this case, also different sensor locations are possible. The foot, however, is mostly used for sensor attachment. Some approaches even attach sensors at both feet and use data fusion [28]. The authors of [24] summarizes all integration approaches with following step sequence. At first, a start and end point of the gait cycle is defined. Note that often the foot flat event is chosen. The next step is about identifying the orientation of the sensor with respect to the global coordinate system. When this relation is known, the accelerometer measurements are projected into the global coordinate system. After removing gravity from the measurements, the accelerometer data is integrated to yield

velocity. Every time the gait cycle starts anew, integration and orientation with respect to the global coordinate system are reseted.

2 Supervised Machine Learning

Machine learning aims at recognizing patterns in a given data set. Typical approaches consists of two subsequent phases: During the first phase, called learning phase, patterns are identified and condensed into a mathematical model. In the second phase, the model is applied to make predictions for unseen inputs. Note that also online algorithms exist, where the learning phase can also take place during operation. Examples for the usage of machine learning are computer vision tasks like face detection or the control of complex systems like autonomous cars or robots. A more concrete example given by Vijayakumar et al. demonstrates the benefits of machine learning [29]. They use a method called locally weighted process regression (LWPR) to learn the inverse kinematics for their 30-DOF humanoid robot. The learned model outperforms the analytical one, because the robot is too complex for an accurate analytical model. So, machine learning techniques especially pay off when dealing with a large amount of data. In such cases, it is hard for humans to design hard-crafted decision rules or even to extract meaningful information. Concluding, the benefits of machine learning are perfectly summarized with following quote from Hal Varian "The ability to take data — to be able to understand it, to process it, to extract value from it, to visualize it, to communicate it — that's going to be a hugely important skill [...]. " [30]

Machine learning can be divided in three application types, namely supervised, unsupervised and reinforcement learning [31]. Here only supervised machine learning is considered. The differences between the application types are constituted by the input data used for learning. In supervised machine learning the learned mathematical model is a input-output mapping or rather a function. The training dataset consists of n observations and is denoted as $D = \{(\mathbf{x_i}, y_i)|i = 1, \ldots, n\}$, where $\mathbf{x_i}$ is a d-dimensional input vector and y_i is the corresponding scalar output value [31]. It can be differed between discrete output categories and and continuous outputs values. In case of discrete categories, the learning task is called classification problem. An example for a classification problem is the recognition of hand-written digits, e.g. for automated zip code identification. In this example, the discrete output categories are all possible digits $y_i \in \{0, 1, 2, \ldots, 9\}$ [31]. If the outputs are continuous, $y_i \in \mathbb{R}$, it is a regression problem. Examples for regression problems are polynomial curve fitting or model estimation for robot control as performed in [32].

The learned function must generalizes to unseen input data, since it is used for future decisions. The generalization ability, however, cannot be estimated with the prediction error on the training set. The learned function might remember the training set perfectly, but perform poorly on unseen data. This case is known as overfitting and is often caused by a too complicated underlying model. To evaluate the generalization ability a test dataset is used [31]. The test set is distinct from the training set and is denoted with D_*.

The problems of gait percent, speed and gait prediction are tackled with respect to the current standard techniques for supervised machine learning. The remainder of this section introduces the used techniques. First, Section 2.1 describes Gaussian process regression. Afterwards, classification principles like support vector machines and multi-class classification are introduced in Section 2.2. In the end, general input-output setups for regression and classification are presented.

2.1 Gaussian Process Regression

Regression aims at learning a function f that can be used to predict function values for unseen inputs. To learn such a function, a training set $D = \{(\mathbf{x_i}, y_i)|i = 1,\ldots,n\}$ is required. It consists of n observations, where $\mathbf{x_i}$ is a d-dimensional input vector and y_i is the corresponding function value. Concatenating the training inputs into a $d \times n$ matrix and the outputs into a vector results in a more compact representation of the training set $D = \{X, \mathbf{y}\}$ [33]. The learned input-output mapping is of the form $y_i = f(\mathbf{x_i}) + \varepsilon$, where ε is independent, identically distributed Gaussian noise with zero mean and variance σ_n^2 [33]. When the function is used to make predictions for unseen data, the input is called test input and is denoted with $\mathbf{x}_*$. The output of the prediction is called and denoted accordingly. In the case of predictions for several test points, the test inputs can be concatenated into the matrix X_* and the corresponding outputs into the vector $\mathbf{y}_*$.

Gaussian process regression belongs to the category of Bayesian non-parametric regression. Instead of learning specific parameters of a function, e.g. the coefficients of a cubic function, it integrates over all possible parameters. Hence, predictions do not depend on the parameters but only on the data points itself. In the following, Gaussian process (GP) regression is defined according to Rasmussen [33].

A GP is a distribution over functions,

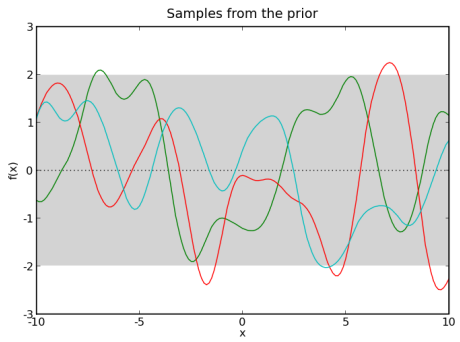
$$f \sim \mathcal{GP}(m(\mathbf{x_p}), k(\mathbf{x_p}, \mathbf{x_q})), \tag{2.1}$$

which is specified by a mean $m(\mathbf{x_p})$ and covariance function $k(\mathbf{x_p}, \mathbf{x_q})$. Because of convenience, the mean function is typically set to zero. However, it is also possible to use the mean function to encode prior knowledge. The covariance function is needed to compute the covariance matrix, so that it determines the properties of all possible functions specified by the GP. Informally, a GP can be described as Gaussian distribution for functions. More formally, it is defined as collection of random variables, from which every collection of variables has a joint Gaussian distribution. In the context of Gaussian processes, a random variables is identified by the function value $f(\mathbf{x})$ at location $\mathbf{x}$.

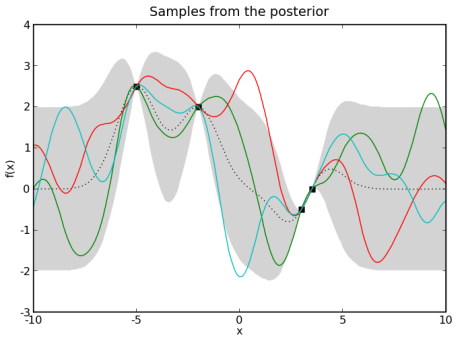
The Gaussian process framework can be used for inference. Starting with the prior which is given by the GP specified by Equation (2.1). The prior encodes all possible functions before observing any training data. When incorporating the training data, the possible functions are reduced to those functions that pass through the training points. The resulting distribution of functions is known as posterior. The mean of the posterior is used to make predictions and its variance is a measure for the prediction's uncertainty. An graphical example is given in Figure 2.1. Regarding this example, Figure 2.1a illustrates samples drawn from the prior and Figure 2.1b shows the prior conditioned on 4 training points (posterior).

Since a GP is a collection of random variables, it is possible to specify the joint distribution of training and test outputs

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \sigma_n^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right), \tag{2.2}$$



(a) Samples drawn from the prior. Figure taken from [34]



(b) Posterior. Figure taken from [34].

Figure 2.1.: (a) shows some samples drawn from the prior distribution specified by $GP(0,0)$. **(b)** depicts samples from the posterior which is the prior conditioned on four training points. The confidence, shown in both plots, is a measure for the uncertainty and is defined as $\bar{x_*} \pm \sqrt{cov(x_*)}$. Close to a training point the confidence is small, since all possible functions must pass through this point. When moving further away from the training points, the function's variability increases and consequently the confidence grows, as well.

where f_* is the predictive distribution of the test outputs and $K(X,X')$ denotes the covariance function evaluated between all points contained in X and X' [33]. The posterior is simply computed by conditioning the joint distribution on the training points

$$\mathbf{f}_*|X,\mathbf{y},X_* \sim \mathcal{N}(\bar{f}_*, cov(f_*)), \text{ where} \tag{2.3a}$$

$$\bar{\mathbf{f}}_* = K(X_*,X)[K(X,X)+\sigma_n^2 I]^{-1}\mathbf{y}, \tag{2.3b}$$

$$\text{cov}(\mathbf{f}_*) = K(X_*,X_*) - K(X_*,X)[K(X,X)+\sigma_n^2 I]^{-1}K(X,X_*). \tag{2.3c}$$

Predicting is done by evaluation Equation (2.3b) and the uncertainty of the prediction is given by Equation (2.3c) [33]. Considering this equations, the relevance of the covariance function becomes apparent.

As described earlier, the covariance function is responsible for the prior or rather the properties of all possible functions. Different covariance functions and combinations have been proposed. The most common covariance function is the squared exponential covariance function

$$k(\mathbf{x_p},\mathbf{x_q}) = \sigma_f^2 \exp(-\frac{1}{2}(\mathbf{x_p}-\mathbf{x_q})^\top M(\mathbf{x_p}-\mathbf{x_q})) + \sigma_n^2 \delta_{\text{pq}}, \tag{2.4}$$

where δ_{pq} is the Kronecker delta which equals one if $p = q$ and is zero otherwise. The remaining parameters are called hyperparameters and consists of the signal variance σ_f^2, the noise variance σ_n^2 and the characteristic lengthscales $M = diag(\mathbf{l})^{-2}$ [33]. Note that the i-th element of l describes how far we need to move in the i-th dimension of the input space to perceive a change. The larger the i-th lengthscale the farer we need to move to observes changes and, consequently, the i-th dimension becomes less important for performing predictions. Different hyperparameters result in different function properties and consequently

cause different predictions. In case of Gaussian process regression, learning can be identified with hyperparameter determination. Only good hyperparameters result in good predictions for unseen data. Since the squared exponential covariance function is infinitely differentiable, it favors smooth functions. Such an assumption can be unsound for physical processes. The Matérn covaraince function is often used as an alternative [33].

A common approach to determine the hyperparameters is to maximize the log marginal likelihood

$$\log p(\mathbf{y}|X) = -\frac{1}{2}\mathbf{y}^\top [K(X,X) + \sigma_n^2]^{-1}\mathbf{y} - \frac{1}{2} log|K(X,X) + \sigma_n^2| - \frac{n}{2} log\, 2\pi. \tag{2.5}$$

Regarding its equation, it can be seen that the log marginal likelihood automatically performs a trade off between model fit and complexity. The first term of eq. (2.5) addresses data fit, the second term introduces a complexity penalty and the last term is a normalization constant.

2.2 Support Vector Machines

Already in 1995 Vapnik introduced support vector machines (SVM) based on the statistical learning theory. After some years support vector machines established as standard for classification. The core idea of SVMs is to enforce the smallest generalization error by formulating the problem of binary classification as quadratic optimization problem. In doing so, the input data is transformed in a higher dimensional space where an optimal separating hyperplane is defined. The solution is sparse, since the hyperplane depends on a subset of training points called support vectors. Support vectors are those points closest to the hyperplane. To classify new inputs, it is sufficient to check on which side of the hyperplane the input resides [35].

In the following support the SVM formulation is introduced according to [31]. The training set is denoted as $D = \{(\mathbf{x_i}, y_i)|i = 1,\ldots,n\}$, where $\mathbf{x_i}$ are d-dimensional input vectors and $y_i \in \{-1,1\}$ are the corresponding binary class labels. Note that the class labels are also known as target values are in [31] denoted with t_i. The hyperplane that linearly separates the training data is given by

$$d(\mathbf{x}) = \mathbf{w}^\top \sigma(\mathbf{x}) + b = 0, \tag{2.6}$$

where $d(\mathbf{x})$ is called decision boundary, $\mathbf{w}$ is the d-dimensional adjustable weight vector, b is the bias and $\sigma(\mathbf{x})$ maps the input data into another, often high dimensional space [31]. If the input data is initially not linearly separable, the mapping may introduce a space where it is. The sign of the decision boundary is used to classify new input data which is denoted with $\mathbf{x}_*$. If $d(\mathbf{x}_*) > 0$, $\mathbf{x}_*$ lies on the left of the decision boundary it is classified as $y_* = 1$. Otherwise it is on the right of the decision boundary and is classified as $y_* = -1$. To achieve the smallest generalization error, the concept of the margin is introduced. The margin is set as the smallest perpendicular distance between the decision boundary and the nearest points of the training set. So, that their is an identical margin on both sides of the decision boundary. According to the definition of the margin, all other training points are located on or behind the margin. Each point on the margin is a support vector. For a graphical illustration see Figure 2.2.

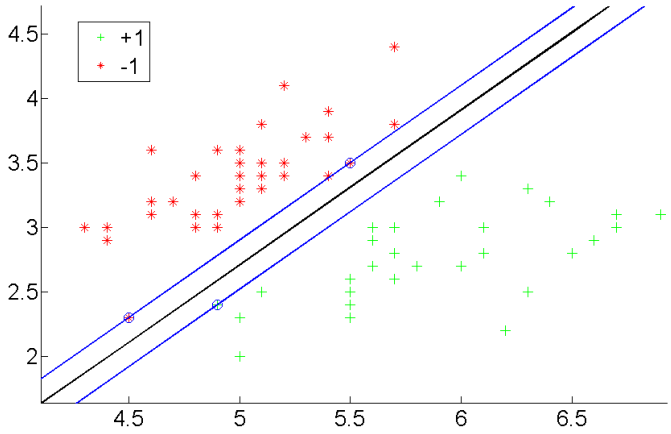


Figure 2.2.: Graphical illustration of the hard-margin support vector machine formulation. The red and green points represent the different class labels involved in support vector machine training. For the given training points, the optimal decision boundary, also known as optimal separating hyperplane, is given by the black line. The black line is chosen, since it maximizes the perpendicular distances to the margins which are denoted by the blue lines. It becomes clear that the margins and consequently the decision boundary depend only on the encircled points lying on the margin. Those encircled points are called support vectors.

For optimization purpose, Equation (2.6) is rescaled so that for points on the margin holds

$$\begin{cases} \mathbf{w}^\top \sigma(\mathbf{x}) + b = 1 & \text{if } y_i = 1 \\ \mathbf{w}^\top \sigma(\mathbf{x}) + b = -1 & \text{if } y_i = -1 \end{cases} \Leftrightarrow d(\mathbf{x}) = y_i(\mathbf{w}^\top \sigma(\mathbf{x}) + b) = 1. \tag{2.7}$$

Consequently, the training data is linearly separable if following condition, given in [31], is satisfied

$$d(\mathbf{x}) = y_i(\mathbf{w}^\top \sigma(\mathbf{x}) + b) >= 1, \quad i = 1, \ldots, n. \tag{2.8}$$

To achieve the smallest generalization error, the margin is maximized. The margin or rather the distance between both margins is computed as difference between the nearest points of both margins (left & right) [35]: $\min_{\mathbf{x}:y_i=1} x\mathbf{w}/|\mathbf{w}| - \min_{\mathbf{x}:y_i=-1} x\mathbf{w}/|\mathbf{w}| = 2/|\mathbf{w}|$. So we must maximize $\|\mathbf{w}\|^{-1}$ which is equivalent to

$$\underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}\|^2, \tag{2.9}$$

subject to Equation (2.8). Casting Equation (2.9) to the Lagrangian formulation leads to

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} a_i\{y_i(\mathbf{w}^\top \sigma(\mathbf{x_i}) + b) - 1\}, \tag{2.10}$$

where a_i are the Lagrangian multipliers [31]. By derivating equation 2.10 with respect to $\mathbf{w}$ and b and using those results we obtain the Wolfe dual [31].

$$L(\mathbf{a}) = \sum_{i=1}^{n} a_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j y_i y_j k(\mathbf{x_i}, \mathbf{x_j}) \tag{2.11}$$

which is maximized with respect to

$$a_i \geq 0, \quad i = 1, \ldots, n \tag{2.12a}$$

$$\sum_{i=1}^{n} a_i t_i = 0. \tag{2.12b}$$

The decision boundary or rather rule can be written as

$$d(\mathbf{x}) = \sum_{i=1}^{n} a_i t_i k(\mathbf{x}, \mathbf{x_i}) + b \tag{2.13}$$

In Equations (2.11) and (2.13) $k(\mathbf{x_i}, \mathbf{x_j}) = \sigma(\mathbf{x_i})^\top \sigma(\mathbf{x_j})$ is called kernel function. It is easier to define $k(x_i, x_j)$ instead of $\sigma(x)$, since the space where the data is linear separable is usually unknown. This procedure is known as kernel trick. The most common kernel function for SVMs is the radial basis function (RBF) which transforms the input data into an infinite dimensional space. The RBF is given by

$$k(x_i, x_j) = \exp\left(-\frac{\|\mathbf{x_i} - \mathbf{x_j}\|^2}{2\sigma^2}\right), \tag{2.14}$$

where σ is a free parameter [36].

Equation (2.11) is a constrained optimization problem that satisfies the Karush-Kuhn-Tucker conditions. This conditions imply that $a_i \neq 0$ only for points lying on the maximal margin ($y_i d(\mathbf{x_i}) = 1$). Since only the points called support vectors are relevant for making predictions for input data (see Equation (2.13)), the solution is sparse.

Because of high computation and memory requirements, solving Equation (2.11) is analytically infeasible. Instead numerical methods are used. The current standard algorithm for learning SVMs is called sequential minimal optimization (SMO) [37]. SMO outputs $\mathbf{w}$ and the Lagrangian multipliers. b is computed by averaging over all points that satisfy $y_i(\mathbf{w}^\top \sigma(\mathbf{x}) + b) = 1$.

The input data is not always linearly separable in kernel space. In this case, the presented SVM formulation, also known as hard-margin SVM, may lead to poor generalization performance. This can be overcome by soft-margin SVMs which allow the misclassification of some training points. Therefore, a slack variable ξ_i is introduced for each training point. Each slack variable represent the misclassification penalty of its corresponding training point. If x_i is classified correctly $\xi_i = 0$, otherwise ξ_i is set to $|y_i - d(\mathbf{x_i})|$ [31]. Consequently, $\xi_i > 1$ if the training point is misclassified and $0 < \xi_i < 1$, if the training point is classified correctly

but is in between decision boundary and margin. Considering the slack variables, Equation (2.8) is reformulated as following condition

$$d(y_i(\mathbf{w}^\top \sigma(\mathbf{x_i}) + b) >= 1 - \xi_i, \quad i = 1, \ldots, n. \tag{2.15}$$

So, we minimize

$$\underset{\mathbf{w},b}{\operatorname{argmin}} C \sum_{i=1}^{n} \xi_i + \frac{1}{2}\|\mathbf{w}\|^2, \tag{2.16}$$

subject to Equation (2.15) instead of Equation (2.9), where C is the trade off between model complexity and training error [31]. If $C \to \infty$, no training error is allowed so that the soft-margin approach equals the hard-margin formulation. The sum of all slack variables (ξ_i) is an upper bound for the misclassification rate, because for misclassified points holds $\xi_i > 1$. The Lagrangian for the soft-margin SVM is given by

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} a_i\{y_i(w^\top \sigma(\mathbf{x_i}) + b) - 1 + \xi_i\} - \sum_{i=1}^{n} \mu_i \xi_i, \tag{2.17}$$

where a_i and μ_i are Lagrangian multipliers [31]. The corresponding Wolfe dual is

$$L(\mathbf{a}) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j y_i y_j k(\mathbf{x_i}, \mathbf{x_j}) \tag{2.18}$$

subject to

$$0 \geq a_i \geq 0, \quad i = 1, \ldots, n \tag{2.19a}$$

$$\sum_{i=1}^{n} a_i t_i = 0. \tag{2.19b}$$

Predictions are still made with Equation (2.13).

To evaluate the performance of a binary classifier, performance measures such as accuracy, precision and recall can be used [38]. In case of active ankle control, predictions should be very precise, because every wrong prediction might influence the prosthesis's user. Consequently only classifiers that achieve small misclassification rates are of interest. The performance measure accuracy defines the overall amount of correct classifications and is given by the number of correct classifications divided by the total number of classifications. Achieving a high accuracy is in our context desirable, because it implies a small misclassification rate.

2.2.1 Probabilistic Predictions

SVMs provide only a classification decision, but no probabilistic outputs. Probabilities can be introduced by using a logistic sigmoid in addition to the trained support vector machine. For more details see [39].

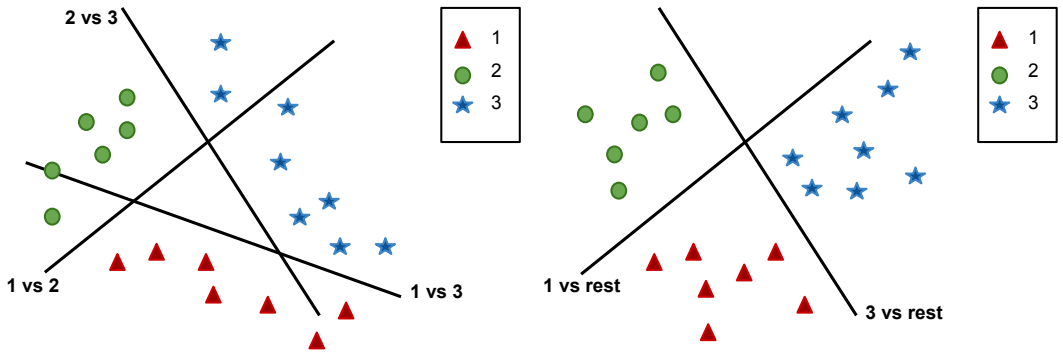
2.2.2 Multi-class Classification

SVMs achieve only binary classification. There are, however, two common approaches to use binary classifiers like SVMs for multi-class classification. In the following, k is the number of different classes involved in multi-class classification. The first approach is called one-versus-the-rest [31]. Here, k classifiers d_i are learned. Classifier d_i uses the data of class i as positive examples ($y_i = 1$) and the data of all other $k-1$ classes as negative examples ($y_i = -1$). To classify new input data, all classifiers need to be evaluated. For input data of class i, ideally only d_i evaluates to 1. If other classifiers evaluated to 1 as well, we need to determine the most probable one. One approach is to choose the class with the maximal decision value $d(x) = max_i d_i(x_*)$. However, the classifiers might have different scales, if they are trained on different tasks [31]. If probabilistic outputs are used in addition, is also possible to chose the most probable classifier.

The second approach is called one-versus-one. Here, a support vector machine is trained for each possible combination of binary classes (1 vs $2, \ldots, 1$ vs $k, \ldots, k-1$ vs k). All in all $k(k-1)/2$ SVMs are trained [31]. To classify new input data, all trained SVMs are evaluated. The classification decision is made according to a majority voting.

Both approaches may involve ambiguities as illustrated by an graphical example given by [31] and shown here in figure 2.3. A problem of the one-versus-the-rest approach might be that the training data is imbalanced, meaning it contains more negative than positive examples. In case of a large number of classes, the one-versus-one approach might be slower in learning and predicting. Their are more advanced methods to address these issues. However, usually both methods are applicable in practice [31].

If the classifiers are equipped with probabilistic outputs, it can be interpolated between the classes. By doing so, values in between two classes, e.g., for class transitions, can be obtained.



(a) One-versus-one approach. Figure adapted from [31].

(b) One-versus-the-rest approach. Figure adapted from [31].

Figure 2.3.: (a) example for ambiguities in case of one-versus-one multi-class classification. The ambiguous region is the white triangle constructed by all one-versus-one classifiers. For inputs of this region, several classifiers might indicate classification at a time. Also for one-versus-the-rest approach, ambiguities can happen. Panel (b) shows a scenario where two of overall three decision boundaries are shown. Already for those two decision boundaries, the white region at the top is ambiguous. Both shown figures are adapted from [31].

When k is the overall number of classifiers, the interpolation rule or rather the expected value [40] is given by

$$y_*^i = \sum_{i=1}^{k} y_i p_i(x_*), \tag{2.20}$$

where y_i is the output label of the i-th classifier, $p_i(x_*)$ the probability of the corresponding classifier given the input data x_* and y_*^i the interpolation result.

2.3 Input-Output Setups

The performance of supervised machine learning directly depends on the used input and output data and their relationship. For example, learning inverse models can be problematic because their relationship often includes redundancies. In case of active ankle control, the input data can be an arbitrary selection of input signals from the prosthesis's sensor information. The output data depends on the problem we want to solve. For gait percent prediction, for instance, the output is gait percent or some data which allows to infer gait percent. Besides the raw input data, it can be differed between some more general input-output setups. While introducing these approaches, the learned predictor, which is either a Gaussian process or a support vector machine, is denoted as function f.

The first approach is called window-in-time setup

$$y_i = f(x_{i-\tau} : x_i), \tag{2.21}$$

where y_i is the prediction and x_i the input at time step i. This approach comprises three tunable parameters, namely the output data of our prediction, the input data and how many previous measurements of the input data (τ) are used. Note that τ is also called window size.

A second approach, named recurrent setup, is given by

$$y_i = f(x_{i-\tau} : x_i, y_{i-1}), \tag{2.22}$$

where all variables have the same meaning as in eq. 2.21. The difference with respect to the window in time approach is that the last gait percent prediction is also considered as input. This gives additional knowledge to the predictor but also requires some accuracy concerning y_{i-1}. If y_{i-1} is not accurate enough, the predictor might be fooled.

3 Motion Capturing Experiments

The supervisory control comprises three predictive tasks, namely gait, speed and gait percent prediction. As input for each predictive task, the prosthesis provides its online sensory measurements. The measurements are obtained with a frequency of 100 Hz and consist of gyroscope and biaxial accelerometer values. Hereby, the gyroscope determines the shank velocity and the accelerometer obtains the acceleration with respect to a coordinate system fixed at the shank. Additional input values like the shank angle can be useful but must be determined numerically or by other means. Each predictive tasks should be implemented with the use of the supervised machine learning techniques described in Chapter 2. Consequently, a task-specific training set, consisting of n inputs and corresponding outputs, is needed to learn a model for each task. In addition to the training sets, test sets are needed to evaluate the performance of the learned models.

This section introduces supervised machine learning for active ankle control and investigates its applicability. Here, each predictive task is analyzed with respect to different setups and input signals. Therefore, the data recorded by a motion capturing system is used. This data is ideal for analytic purposes, since it is really accurate and noise free. The analysis results in the identification of sensors that are good predictors for each task. Afterwards, the structure for a supervisory controller is proposed. Note that Section 4 is about proofing this concepts with real sensor data.

The remainder of this chapter is organized as follows. Section 3.1 describes the creation of a motion capturing database and how to process the collected data to obtain the required information for training and test sets. In Section 3.2, performance objectives for evaluating learned predictors are introduced. Each of the the next three subsections is dedicated to the analysis of one of the predictive tasks. The chapter concludes with a summary and the identification of a control structure in Section 3.6.

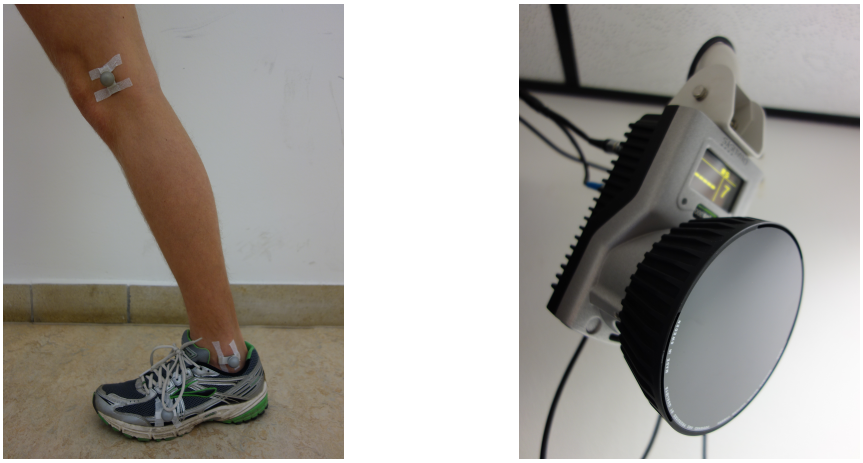
3.1 Database Generation

To facilitate the implementation of the supervisory controller, a database is created. The database is used to generate training and test sets for the tasks of gait, speed and gait percent prediction. Moreover, it is possible to generate different setups for each task, so that the most efficient one can be determined.

Before creating the database itself, the functionality of the supervisory controller or rather the prosthesis needs to be clarified. This is important since the input and output data for each task must be know for training and test set generation. First, the prosthesis should be able to classify the intended gait. Here, we only discriminate between walking and running.

Table 3.1.: Walking and running data contained in the motion capturing databsae

Gait	**Speeds** $[m/s]$
Walking	0.5, 1.1, 1.6, 2.1, 2.6
Running	0.5, 1.1, 1.6, 2.1, 2.6, 3.0, 4.0



(a) Motion capturing setup. **(b)** Motion capturing camera.

Figure 3.1.: (a) marker setup for motion capturing. Markers are placed at knee, ankle and little toe, respectively. (b) one of the high speed cameras used for motion capturing.

Secondly, the speed of the user needs to be inferred. As listed in Table 3.1, different speeds for walking and running are considered. Thirdly, it must be possible to predict gait percent for all gait-speed combinations. The term gait percent refers to the phase of ankle movement within a gait cycle. At heel strike of the prosthesis, gait percent is 0%. During gait cycle, gait percent increases continuously until it reaches 100%. With the next heel strike of the same leg gait percent drops to 0% and starts all over again.

In order to train and test the functionality described above, the database contains samples for each scenario described in Table 3.1. All samples were recorded by one test subject on a treadmill, whereat the test subject was a healthy male person of 1.86 m height and 76 kg weight. The recorded database consists of motion capturing data and the data of two force plates. In the case of motion capturing, a QUALISYS system recorded the positions of passive markers attached at both ankle and knee joints. The setup of one leg is displayed in Figure 3.1a and one of the high speed cameras of the QUALISYS system is shown in Figure 3.1b.

Regarding the force plates, each of the two measures the impact of one foot (left and right) on the treadmill.

The recordings for a specific scenario, e.g. walking at 0.5 m/s, were performed as depicted in Figure 3.2.

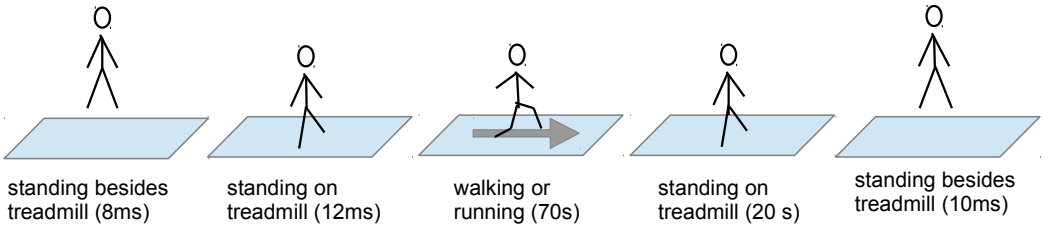


Figure 3.2.: Recroding procedure for each considered gait-speed combination. In a first step the system is callibrated. Afterwards, the subject stands on the treadmill to record a weight capture and a fixed reference setup. After executing the considered gait-speed combination, the same procedure is repeated the other way around (standing on treadmill followed by standing besideds treadmill).

From each recording, distinct, fixed-size intervals are used as basis for extracting training and test samples. This basis data, however, is motion capturing and force plate data. Instead, data as it would be perceived by the sensors is required. Only when using sensor-like data, we get a sufficient test bed for the above presented machine learning methods. In this case, the results can probably be transfered to the real prosthesis, because all used data is directly or indirectly inferable from the prosthesis's sensors. For the used prosthesis shank angle, shank angular velocity and acceleration in x- and y-direction are available. The corresponding sensor values are, expect from shank angular velocity, displayed in Figure 3.3. In addition it would be possible to equip the prosthesis with a force sensor.

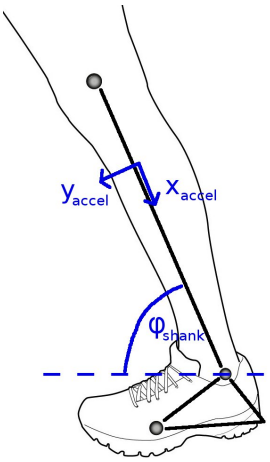


Figure 3.3.: Sensor values provided by the prosthesis.

The transformations form measured data to sensor data are described below.

Shank angle. The motion capturing system records the knee and ankle markers. Using these points and the intersection point of the line parallel to the ground and passing through the ankle and the line perpendicular to the ground and passing through the knee, a right triangle can be constructed. If the knee is positioned before the ankle , right triangle geometry gives the shank angle as $\theta_{\text{shank}} = sin^{-1}(a/c)$. In the remaining cases, the shank angle is computed as $\theta_{\text{shank}} = 180° - sin^{-1}(a/c)$.

Shank velocity. The shank angle can be computed for two consecutive time steps and the frame rate of the motion capturing system is known. So, the shank velocity can be computed with the use of the difference quotient

$$\dot{\theta}_{\text{shank}} = \frac{\theta_{\text{shank},t+1} - \theta_{\text{shank},t}}{\text{fr}},$$

where fr is the frame rate and $\theta_{\text{shank},t}$ the shank angle at time step t.

Accelerometer data. Using the motion capturing data, it is possible to determine the linear acceleration of the point where the accelerometer is mounted. This accelerations are relative to a coordinate frame with the same orientation as the motion capturing coordinate frame. The accelerometer frame, however, rotates with the shank. So, the first step is to transform the linear acceleration into the accelerometer frame. Since the accelerometer measure gravity in addition, gravity needs to be subtracted from the acceleration in motion capturing frame orientation.

Above transformations reconstruct the online sensory data or rather the inputs for the predictive tasks. In supervised learning, the output must be known as well. For each predictive task, the output is reconstructed as follows.

Gait. Gait is known for each recording.

Speed. The speed is imposed and recorded by the treadmill.

Gait Percent. Gait percent is determined by discretizing from one heel strike to the next heel strike of the same leg. Starting with 0% at heel strike and increasing in equidistant steps until 100% is reached at the last measurement before the next heel strike of the same leg. To identify when the heel strikes of both legs occur, the force plates are used.

In the following, predictors with different input signal combinations are used and compared. To facilitate a more concise communication some abbreviations for the signals are used. Shank angle and shank velocity are abbreviated with *angle* and *vel*. For the acceleration signals in x and y direction *Ax* and *Ay* are used, respectively. Together shank angle and shank velocity are referred to as shank data and both acceleration signals together are termed accelerometer data.

3.2 Performance Objectives

Before evaluating the three different predictive tasks, performance objectives valid for each of those are given. Here, three main objectives such as accuracy, speed and a small number of input signals are important. In the following each objective and its relevance for active ankle control is introduced shortly.

Accuracy

Accuracy is about achieving a high prediction quality or rather correctness. All predictions must be as exact as possible, since they directly influence control and consequently the user of the prosthesis. In the worst case, wrong predictions aggravate the interplay between user and prosthesis or cause the user to stumble or even to fall. Accurate predictions, in contrast, facilitate an active user support and improve locomotion. In more detail, the motion of an intact limbs is mimicked more closely and deficiencies introduced by passive devices are overcome. To achieve good support, we define for each predictive task how accurate it should be. Gait percent prediction is only applicable if the prediction error is in between ± 10 gait percent. Errors of 10% will cause no big troubles, but might be noticed by the user. Especially in the stance phase, e.g. at prominent events like hell strike or push off, such errors can become evident. Regarding the swing phase, errors are not that noticeable, since it is only important to reposition the foot for the next heel strike. To increase user confidence, gait percent errors must be not perceived by the user. Therefore, the error during the stance phase, which lasts from 0 to 60 gait percent, should be bounded at least to ± 6 gait percent.

Regarding the considered speeds for speed prediction (compare Table 3.1), two consecutive speeds are separated by $\pm 0.4 m/s$. Speed prediction must only be wrong by the distance of two consecutive speeds, since larger errors will result in too big differences between intended and predicted speed. Ideally, the error should be even lower. If the error is below $0.2 m/s$, it is additionally possible to match the prediction with exactly one of the speeds given in Table 3.1. In this case, speed would always be classified correctly. When classification is used for speed prediction, the performance measure "accuracy" should be as high as possible. For approving a predictor as sufficient good for active ankle control, we define a threshold of 90% accuracy. When exceeding this threshold, the rate of misclassification is rather small so that the user is not influenced that often. Furthermore, we still require that a misclassified speeds is just one class above or below the real class. Note that high accuracies in case of speed, and also gait prediction, do not cause too specialized models with substandard generalization ability. Such models are especially prevented by the following properties of

gait: Walking and running are both of repetitive nature and are unique for each subject. Due to the repetitive nature, the patterns produced for a given speed and gait are similar over time. In [41] it is observed that stride-to-stride fluctuations are normally relatively small and that gait parameters e.g., stride time, vary by just a few percent. Moreover, the locomotion system is reported to shows fractal-like properties as also observed for the heart rate. When fractal-like systems show perceivable deviations from normal behavior, this can be an indicator for certain diseases [41]. The uniqueness requires to fine tune the prosthesis for each user. Note that this ensures optimal performance and is required for passive prosthesis as well. Nevertheless, generalization along subject is a topic for future work. In case of gait prediction, the same threshold as for speed prediction is used.

Speed

Speed is about the computation time required to perform predictions and also about delay. For analyzing prediction time, we examine the execution of one gait cycle. Depending on speed and gait, the recorded gait cycles typically takes about 0.6 to 1.1 seconds. The supervisory controller must perform all its computations within this time. If this does not hold, the controller is unusable. Since the supervisory controller consists of gait, speed and gait percent prediction, we must consider all these tasks. If the tasks are performed after each other, each predictor is allowed to require approximately $(1/3)\cdot 0.5$ seconds for predicting one gait cycle. When some predictors can operate in parallel, the time for each predictor increases. Note that all computations performed in this thesis were executed on a laptop computer running MATLAB. The laptop computer is equipped with a Intel Core 2 Duo with 2,40 GHz and 4 GB ram. Delay has nothing to do with computation time. Instead, it describes how long it takes until changes in motion, e.g. the user changed to a different speed, become visible in the predictor's outputs. The learned predictors either use the window-in-time or the recurrent setup. Both setups can tune the window size (τ) to achieve optimal performance. With increasing window size, more previous sensor measurements are stored and used as prediction input. In case of $\tau = 1$, for example, only the actual sensor measurement is considered, whereas for $\tau = 5$ the last 4 sensor measurements are used in addition. If abrupt changes in motion occur, the predictors are confronted with some previous measurement recorded before the change in motion and only one measurement recorded afterwards. Based on this input data, the predictors might detect the change not immediately, since the old recordings still influence the predictions. With future sensor measurements, the old data becomes less and the chances for detecting the change increase. The change in motion, is at the latest, obvious to the predictors when all old data is flushed. For a predictor with window size τ it takes $\tau - 1$ steps until all old data is flushed. Consequently, small window sizes are favorable. The predictor with $\tau = 1$ is most favorable, since it guarantees no delay. All other window size might introduce delay.

Data Spareness

Regarding the input data given to each predictor, we require the predictor to use as less input signals as possible. In detail, the input signals depend on the used sensors. The less sensors required, the cheaper and easier the used approach is accomplished. Here, the considered prosthesis is equipped with a biaxial-accelerometer and a gyroscope. In addition, it is possible to buy and mount a force sensor to the prosthesis. This is only done, if force data improves prediction performance appreciable. Note the difference between signal and sensor. A

sensor, e.g. accelerometer, can give more than one signal. Regarding the accelerometer again, acceleration in x and y direction are given. To gain insights about the sensor's contributions to prediction performance, all sensors are evaluated in isolation and in combination with other sensors. If a sensor is evaluated, all signals provided by the sensor are considered as input, e.g. in case of the accelerometer we use acceleration in x and in y direction. Combinations of sensors where just selected sensor signals are used, are not evaluated in detail. This would increases the number of different scenarios and is not really necessary, since we can always access all information a sensor provides. Besides evaluating all sensor combinations, we also evaluate all possible input signals in isolation. In this case, acceleration in x and in y direction are both regarded independently. By evaluating just single inputs, we hope to extract the value of each signal in context of the considered predictive tasks.

With all objectives in mind, we define the importance of each objective and a decision rule. The primary focus is on accuracy, since the prosthesis should support and not hinder its users. Also speed is important for user confidence. If large window sizes are required to achieve high accuracies, delays might be introduced and therefore usability might decrease. In contrast, the amount of sensors attached to the prosthesis is only important from a design point of view. All in all, we choose the smallest window size and amount of input data to achieve at least the required accuracy.

3.3 Gait Percent Prediction

For gait percent prediction we use the Gaussian processes framework. When predicting gait percent, it can be assumed that gait and speed are already known. Both can be determined beforehand using the gait and speed predictors presented in the following sections. Consequently it is sufficient to learn an unique gait percent predictor for each available gait-speed combination. In the following gait percent prediction is analyzed exemplary for walking at 1.6 m/s. Note that it equally scales for all other gait-speed combinations.

Window-in-Time Setup

To analyze gait percent prediction, both input-output setups presented in Section 2.3 are compared. Starting with the window-in-time setup, where the tunable parameters are the output data, the input data and the considered amount of previous measurements (τ). Since gait percent is to be predicted, an obvious approach for the output data is $y_i \in [1, 100]$. The input data is an arbitrary combination of accelerometer, shank and force data. For τ, also known as window size, 1, 5 and 10 are considered. In the following, we investigate the impact of different input data and different window sizes by fixing one of the parameters and varying the remaining one. First, the input data is varied and the window size is fixed to one. None of the input signals alone (Ax, Ay, angle, velocity) is able to constitute a working predictor. In detail, all predictors trained with one of these single input signals and window size set to one, give similar performance as shown in Figure 3.4a. Regarding this figure, the dashed line shows the real gait percent value during one step, the blue line is the prediction and the gray shaded area is the confidence. The shown prediction performance is those of a predictor using only shank velocity as input and performing on the test set. Note that all following evaluations and comparison of course reflect the performance on the test sets.

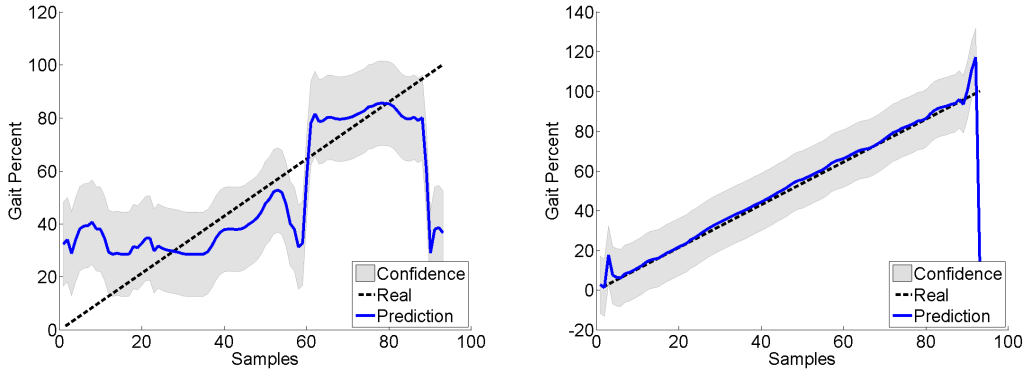
To improve predictions, it is required to have information that distinguish different gait percent values. This can be done by either combining different sensors or by increasing the

Table 3.2.: Lengthscales for the window-in-time predictors with $\tau = 1$. The lengthscales are ordered by the input signal order given in each column heading. Shank angle and shank velocity are abbreviated with *angle* and *vel*. For the acceleration signal in x and y direction *Ax* and *Ay* are used, respectively. The force data of the left foot is denoted with *fzl*. For angle-vel, for instance, 0.14 is the lengthscale corresponding to angle and 0.31 corresponds to vel.

angle	vel	Ax	Ay	fzl
0.22	0.33	0.30	0.28	0.25

angle-vel	Ax-Ay	angle-vel-fzl	Ax-Ay-fzl	angle-vel-Ax-Ay	angle-vel-Ax-Ay-fzl
0.14	0.38	0.49	0.53	0.06	0.13
0.31	0.21	0.23	0.50	0.61	0.80
		0.24	0.49	0.36	0.39
				0.45	0.24
					1.10

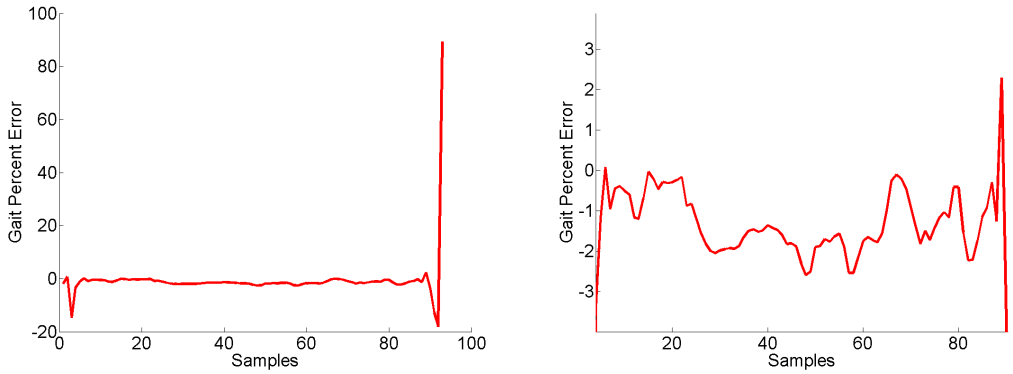
window size. At first, the window size remains fixed and different sensor combinations are evaluated. A promising input combination is shank angle and velocity, which is obtainable from the gyroscope. The corresponding prediction results for one step are shown in Figure 3.4b. Furthermore, Figure 3.5 displays the prediction error. A zoomed in view on the error (left panel) indicates good performance, since the error is in between $\pm 4\%$. Such an error meets the defined acceptance criterion for a final gait percent predictor, but it is not given for the whole prediction area. Considering the whole error (right panel), there are high peaks at the boundaries or rather near 0 and 100 gait percent. Most likely this is due to the large step between 100 and 0 gait percent which occurs at heel strike. In the following, we differ for gait percent prediction between error and boundary error. The term error refers, if not stated otherwise, to the zoomed in error and the boundary error is concerned with the are near 0 and 100 gait percent. Details on how to deal with the boundary peaks are given later. For the other combinations of input signals we obtain the performance shown in the first row of Table 3.3.



(a) $\tau = 1$, input: shank velocity.

(b) $\tau = 1$, input: shank angle and velocity.

Figure 3.4.: Both panles show the gait percent prediction results for walking at $1.6m/s$ when using a window-in-time setup with $\tau = 1$. The difference in prediction performance is constituted by the input provided to the predictors. In panel (a) just velocity is used as input, whereas in panel (b) shank angle and velocity are given.



(a) Overall error. **(b)** Zoomed in view.

Figure 3.5.: (a) shows the gait percent error when using a predictor for walking at $1.6m/s$ with $\tau = 1$ and shank angle and velocity as input. The larger errors near 0 and 100 gait percent are called boundary errors. In panel (b) a zoomed in view on the error is shown. For gait percent prediction we distinguish between error, which is a used as synonym for the zoomed in view, and boundary error.

As described in Section 2.1, each lengthscale used by the covariance function states the relevance of its corresponding input signal. If a lengthscale gets large, is plays no role for Gaussian process inference. In Table 3.2 the lengthscales are denoted for each predictor. The lengthscales are ordered by the input signal order given in each column heading. The results shown in the first row of Table 3.3 confirm the prediction results displayed in Figure 3.4. Single input signals do not constitute good predictors, at least for window size 1. Instead, good prediction results are obtained by using shank angle and shank velocity as input. Using accelerometer or force data in addition does not lead to noticeable improvements. The error bounds decrease at most to 3% to -4% which is nearly the same as for just shank angle and velocity ($\pm 4\%$). The boundary peaks, however, change perceivable. For instance, for combining shank angle and velocity with a force sensor, the boundary peaks decrease to about $\pm 20\%$. This is most likely due to the force sensor which gives more precise information about heel strike. Regarding the lengthscales for gyroscope and accelerometer data in combination, accelerometer data seems a little bit more distinguishing than shank velocity. However, using accelerometer data in addition introduces a second sensor and accelerometer data alone is not sufficient enough as can bee seen by the prediction performance with Ax and Ay as input. So, for a window size of one, shank angle and velocity are the most favorable input signals. They require only one sensor and perform good (expect from the boundary issues).

Another parameter to vary is τ. With increasing τ more previous measurements are considered and the predictor has more data for distinguishing between the inputs. The results are given in Table 3.3.

When increasing the window size, the single input predictors improve the most. While their predictions for $\tau = 1$ are not useful at all, they perform good for $\tau = 5$ or at least for $\tau = 10$. The best single input predictor is obtained when using shank angle as input. The predictor for Ax is not as good as the one for shank angle, but only the error of those two single input predictors is already for $\tau = 5$ bounded to $\pm 5\%$. Please keep in mind that the error and the boundary peaks are treated independently. For combinations of shank data and other input signals, the prediction results do not improve much. Consequently, window size

Table 3.3.: Prediction errors for varying window sizes (τ). The first row for each window size is the error(+/-) in gait percent, whereas the second row is the boundary error (+/-) in gait percent. For the differences between both error types see Figure 3.5. Note that the abbreviations used in each column headings are the same as in Table 3.2. *angle-vel*, for instance, is the abbreviation for shank angle and shank velocity as input.

τ	angle	vel	Ax	Ay	fzl
1	40/40	40/40	42/42	42/42	40/40
	60/40	70/40	80/40	80/60	30/60
5	3/3	10/12	5/5	10/0	10/12
	100/90	60/60	40/40	40/40	60/60
10	2/0.5	3/3	5/4	6/6	3/3
	80/80	60/60	30/30	20/20	60/60

τ	angle-vel	Ax-Ay	angle-vel-fzl	Ax-Ay-fzl	Ax-Ay-fzl Ax-Ay	angle-vel-Ax-Ay-fzl
1	4/4	20/20	4/4	6/7	4/3	4/3
	90/30	45/10	20/25	50/25	40/15	85/50
5	3/3	4/4	2/4	3/3	2/4	4/3
	20/20	10/12	80/40	20/20	5/20	10/25
10	3/3	5/4	2/4	4/3	3/3	3/3
	30/30	10/11	70/50	8/6	25/25	25/25

increases give no additional information here. The boundary peaks, however, change with varying window size. Sometimes the changes are beneficial and sometimes not. All in all, the boundary errors are still too large. In the best case the boundary peaks should decrease to the size of the normal error. For accelerometer data only and accelerometer data in combination with the force sensor, the prediction results improve to the degree of shank angle combined with shank velocity. The combination of shank angle and velocity remains still favorable, since it requires only one sensor and performs good.

As described in Section 3.2, a larger τ may increase the time to notice an abrupt change. Moreover, it takes $\tau - 1$ time steps until all signals are flushed. In case of gait percent prediction, however, changes occur rather continuous than abrupt. Nevertheless, more inputs lead to an increase in computation time. This is because predicting (see Equation (2.3b)) requires the computation of the covariance function between the test points and all training inputs. The input size increases if τ increases or more input signals are used. For fixed τ and different numbers of input signals, the differences in computation time is, however, neglectable. For increasing τ, the computation time increases on average. For $\tau = 1$ the gait percent computation for one step takes $0.0861s$, for $\tau = 5$ and $\tau = 10$ it takes $0.9841s$ and $0.1951s$, respectively. The predictor obtained by using shank angle and velocity as input performs already good for $\tau = 1$. In addition, the small window size gives the benefits of shorter computation time and fast reactions to abrupt changes.

Recurrent Setup

The second input-output setup is the recurrent one. By using the previous prediction as additional input, the predictor is provided with additional knowledge. This knowledge is about explicitly incorporating the linear increasing nature of gait percent. If, for instance, the last prediction is 5%, the next prediction is most likely greater than 5% and less than 10%. This implies that such a predictor will have difficulties with sudden jumps which, however, do not

Table 3.4.: Prediction results for the recurrent setup with $\tau = 1$. Besides the prediction errors in gait percent, the predictor's lengthscales are given. All lengthscales are ordered by the input signal order given in each column heading. For information on the used abbreviations see Table 3.2. The last lengthscale which is gray corresponds to the input from the last gait percent prediction.

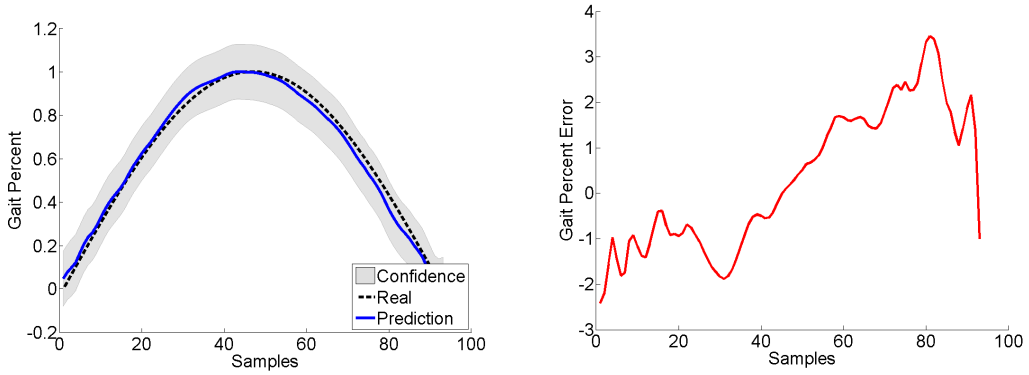
	angle	**vel**	**Ax**	**Ay**	**fzl**
Error (+/−)	3/3	6/4	4/4	4/5	5/4
Boundary Error (+/−)	40/20	60/40	40/40	40/40	20/70
Lengthscales	0.86	0.24	0.42	0.26	8.79
	0.47	0.43	0.73	1.59	0.31

	angle-vel	**Ax-Ay**	**angle-vel-fzl**	**Ax-Ay-fzl**	**angle-vel-Ax-Ay**	**angle-vel-Ax-Ay-fzl**
Error (+/−)	1/3	4/4	5/4	3/4	2/3	3/3
Boundary Error (+/−)	20/20	90/60	30/20	80/40	40/60	45/10
Lengthscales	0.61,	0.70,	0.46,	0.57,	0.27,	0.14,
	0.19,	0.61,	0.11,	0.52,	0.36,	1.24,
	1.35	0.84	21.64,	30.78,	0.38,	0.35,
			0.67	0.78	0.36,	0.27,
					0.90	7.84,
						1.35

occur often. Moreover, it must be assumed that the error of the last prediction is bounded. If this bound is exceeded, the predictor might be fooled. From the window-in-time setup it seems reasonable to assume a bound or rather a prediction error of ±5%. When assuming this bound and fixing the window size to one, we obtain the results presented in Table 3.4. Regarding the results, several things can be noticed. First, all predictors that were unusable for $\tau = 1$ and the window-in-time setup improved to usability when changing just the setup to the recurrent one. The affected predictors are those for the single sensor inputs and for accelerometer data only. In this cases, the last prediction's influence is rather high, leading to the improvements compared to the window-in-time setup. For the combined signals were the predictions were already good, the legthscales reveal that the last prediction's influence is not that high. Here, the additional knowledge of the last prediction does not boost the performance. Second, the last prediction seems to be easier to uses as heel strike indicator than the force sensor. For all predictors where the force sensor is used, the lengthscale corresponding to this sensor gets larger. Instead, the last prediction's lengthscale is small and in use. Third, the boundary problems also remain for the recurrent-approach. Expect for the boundaries, nearly all predictors are limited to a prediction error of at most ±5%. So in this area, the initially assumed bound for the error is met and the predictor is not fooled. If a recurrent predictor is applied the first time, an initialization for the last prediction is needed. One can either use the prediction of a window-in-time predictor or require some initial state for powering and initializing the prosthesis.

Transformed Output

All previous predictions show some high peaks at the boundaries. This is most likely due to the large step between 100 and 0 gait percent that occurs at heel strike. To avoid this large step and to improve the predictions at the boundaries, the output is transformed. Since ankle



(a) Prediction results when transforming the output.

(b) Gait percent error corresponding to (a).

Figure 3.6.: Panel (a) shows the prediction performance for walking at $1.6m/s$ when using a transformed predictor with $\tau = 1$ and shank angel and velocity as input. By transforming the output of the predictor, the boundary errors are eliminated. (b) depicts the overall gait percent error of the corresponding predictor.

movement, and therefore gait percent, is periodic, an obvious choice is to use a sine or cosine signal as prediction output. This leads to two possible transformations for the output signal

- $t(y) = \sin(y')$
- $t(y) = \cos(y')$.

Regarding the possible transformations, valid choices for y' are

1. $y' = \frac{2\pi}{100}y - \pi$
2. $y' = \frac{\pi}{100}y - \frac{\pi}{2}$,

where $y \in [0, 100]$. Option 1 converts y in a range of $[-\pi, \pi]$, whereas option 2 converts to the range $[-\pi/2, \pi/2]$. When using the cosine transformation with option 2 for the input data of shank angle and velocity, a window-in-time approach and a window size set to 1, we obtain the prediction results displayed in 3.6a. The corresponding prediction error is given in figure 3.6b. A comparison of all transformations applied for all sensor combinations with window sizes fixed to 1 is given in Table 3.5. Here, it is not differed between error and boundary error. Instead we give just the overall error. Using the above transformations the output signal is continuous and has no jumps, expect for the sine transformation with $y' \in [-\pi/2, \pi/2]$. For this case, the jump of the output signal is reduced from a range of 100 to a range of 2. This does, however, not help with the boundary issues, as can be seen from the second column of Table 3.5. The large overall errors present in this column are due to the boundary errors. Compared to the untransformed setup, the boundary error is reduced but still present. For angle and velocity as input, the boundary error is about -30 to 90% for the untransformed case and about -20 to 30% for the sine transformation with $y' \in [-\pi/2, \pi/2]$. In all other cases, namely were the transformed output signal does not jump, the boundary peaks are eliminated. The sine and cosine transformations that uses $y' \in [-\pi, \pi]$ perform equally good. In case of sensor combinations, however, the prediction error is not that accurate as for the untransformed case (expect the boundaries). When using

the cosine transformation with $y' \in [-\pi/2, \pi/2]$ instead, the prediction error gets as good as for the untransformed case. So, the predictors using this kind of transformation are favorable.

A remaining problem is that the transformation is not invertible on the whole domain. For instance, the arccos can only map from $[-1, 1]$ to $[0, \pi]$. In contrast, the best performing transformation, the cosine for $y' \in [-\pi/2, \pi/2]$, lives on a different domain. Here, the best performing transformation is used to describe how to invert the output of the transformed predictor. Two equations are needed to transform to gait percent. One equation for the case where the prediction result is still increasing and consequently is located before the maximum turning point ($cos(x) = 1$). A second equation is needed for the decreasing case. Both transformations are given below

$$y_* = \begin{cases} 99\left(\frac{\arccos(y_*^t)}{\pi} + \frac{1}{2}\right) + 1 & \text{if } d = \text{increasing} \\ 99 - 99\left(\frac{\arccos(y_*^t)}{\pi} + \frac{1}{2}\right) & \text{if } d = \text{decreasing,} \end{cases} \tag{3.1}$$

where y_*^t is the output of the transformed predictor, d denotes if y_*^t is increasing or decreasing and y_* is its transformation to gait percent. A second predictor is used to determine the location (increasing/decreasing) on the cosine curve. For the presented case, the second predictor can either be an untransformed predictor or a predictor with a sine output. When the untransformed predictor is used, the decision is made according to

$$d = \begin{cases} \text{increasing} & \text{if } y_*^d \leq 50\% \\ \text{decreasing} & \text{if } y_*^d > 50\%, \end{cases} \tag{3.2}$$

where y_*^d is the output of the untransformed predictor. Even though the untransformed predictor is prone to boundary errors, it is sufficient for making the above decision. Only the heel strike is detected about one sensor sample to early, so that y_*^d drops one sample earlier below 50%. By knowing this fact, it is easy to correct for it. All in all, the boundary issues are eliminated by learning two predictors and computing the prediction result with Equation

Table 3.5.: Comparison of different output transformations. For each transformation($t(y)$), output range and input signal combination the overall error(+/-) in gait percent is given. For the abbreviations used for the input signals see Table 3.2. As input setup we used the window-in-time approach with $\tau = 1$.

t(y) - range	**angle**	**vel**	**Ax**	**Ay**	**fzl**
cos - $\pi/2$	8/15	20/30	30/30	20/30	4/4
sin - $\pi/2$	60/40	80/50	80/60	80/60	40/60
cos - π	30/50	40/60	70/50	80/80	60/60
sin - π	80/20	80/30	90/30	80/40	50/30

t(y) - range	**angle-vel**	**Ax-Ay**	**angle-vel-fzl**	**Ax-Ay-fzl**	**angle-vel-Ax-Ay**	**angle-vel-Ax-Ay-fzl**
cos - $\pi/2$	4/4	10/25	4/4	10/25	4/4	3/3
sin - $\pi/2$	30/20	30/50	90/30	20/20	35/20	40/10
cos - π	7/6	30/50	6/6	20/20	7/7	6/8
sin - π	7/6	50/40	6/8	10/20	8/8	8/8

(3.1). The transformation process is shown in Figure 3.7. Note that this figure includes the results for predicting gait percent for four consecutive steps. Both used predictors, the transformed and the untransformed one, use shank angle and velocity as input and apply a window-in-time setup with $\tau = 1$.

Covariance Function

The input-output setup is not the only subject of change. It is also possible to changes the covariance function used by the Gaussian process. An alternative to the squared exponential covariance function, is the Matérn covariance function which is said to be more realistic for physical processes [33, 42]. Regarding this covariance function class, ν is a positive, tunable parameter. The larger ν, the more often the covariance function is derivatable. If $\nu \to \infty$, we get the squared exponential covariance function [33]. Here, three different classes or rather smoothnesses are considered, namely $\nu = 1/2$, $\nu = 3/2$ and $\nu = 5/2$. For all classes we obtain very similar results as with the squared exponential covariance function. Using for instance angle and velocity as input for an untransformed predictor and fixing τ to one, results for all considered covariance functions in a gait percent prediction error of approximately $\pm 4\%$. Moreover, the boundary peaks are very similar. Regarding the prediction results in detail, it is possible to note that the results get smoother with increasing ν. In some cases smoothness leads to larger errors, since the underlying function cannot vary that fast. The ability to change rapidly, however, can lead to larger error as well. All in all, there are detailed differences but the overall prediction performance is identic.

Speed Independent Gait Percent Prediction

As shown above, we get accurate prediction results for gait percent prediction. Nevertheless it is required to learn and use a predictor for each gait-speed combination. This is, especially when dealing with a large number of speeds and gaits, tedious. Instead, it would be more convenient to have only one gait percent predictor for each gait. Thereby, also the dependencies are reduced. Until now, gait percent prediction depends on gait and speed prediction. If one of those predictors is wrong, gait percent prediction is also wrong. If generalizing over speeds is possible, the predictor only depends on gait prediction and, consequently, the number of sources of defect is reduced. To test the generalization ability, all experiments were conducted again. This time a predictor for all speeds was used instead of

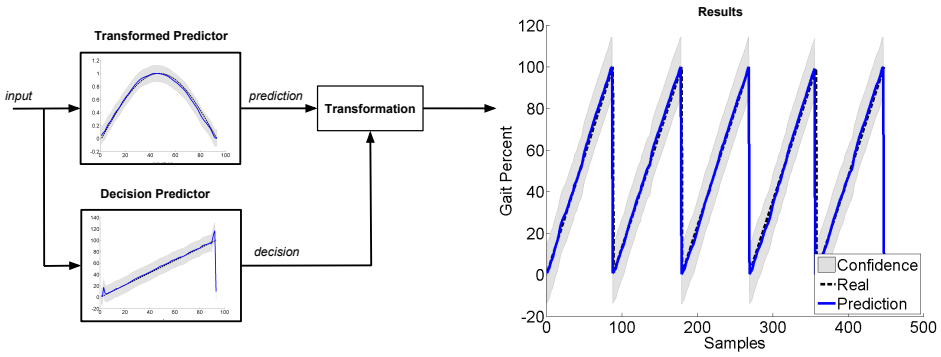
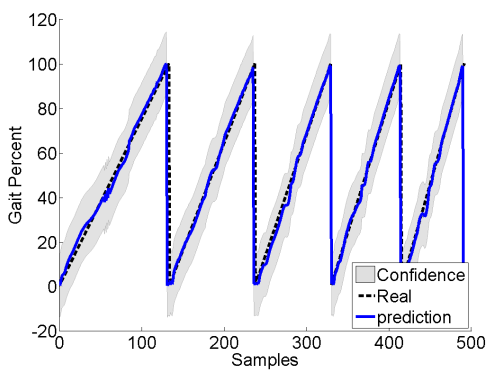
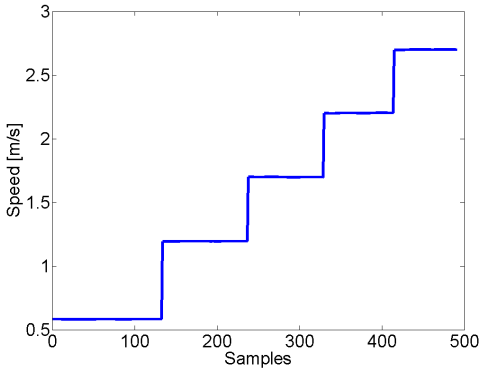


Figure 3.7.: Two predictors are learned. One transformed predictor that gives the prediction results and a second predictor as transformation aid. Depending on second predictor different transformation is applied to achieve gait percent. Results in precise prediction performance, here displayed for 4 consecutive steps.



(a) Gait percent prediction results for different speeds.



(b) The speeds corresponding to the prediction results in Panel (a).

Figure 3.8.: Prediction performance for a gait percent predictor based on Gaussian process regression, shank angle and velocity as input and a window-in-time setup with $\tau = 1$. Panel (a) shows the prediction results for five different steps, where each steps corresponds to a different speed. The precise speeds are given in Panel (b).

a predictor for a special gait-speed combination. The results indicate generalization ability, since the prediction performance is similar for all experiments. More in detail, the prediction error increases only a little when generalizing from one to all considered speeds. For instance, we presented a predictor for walking at 1.6 m/s, using angle and velocity as input, and setting $\tau = 1$, that gives a prediction error of $\pm 4\%$ and eliminates boundary peaks. Using the identical setup, the predictors for all speeds gives an prediction error of $\pm 5\%$ without boundary issues. The prediction performance of this predictor is shown in Figure 3.8a. This figure displays five steps, where each step corresponds to one of the considered speeds. The first step is performed at 0.5 m/s, whereas the the last step is performed at 2.6 m/s. In between the speed increases as denoted in Figure 3.8b. Note that for the example in Figure 3.8, the learned predictor might learn a relationship similar to those for Holgates's tibia based control [11]. More details about the analytical relationship proposed in [11] are given in Section 1.3. If the learned relationship is similar to tibia based control cannot be said for sure and is especially for all predictors with other input setups not true. Since the predictor generalizes for all speeds, it is possible to reduce the number of training data or rather training steps needed for each speed. Using just a few steps for each speed is sufficient and reduces the prediction time for one gait cycle to the time of a specific gait-speed predictor with the same input signals and window size. If the number of used training speeds increases above the number considered here, the prediction time will increase as well. However, not all speeds are required for learning, since the predictor generalizes somehow.

3.4 Speed Prediction

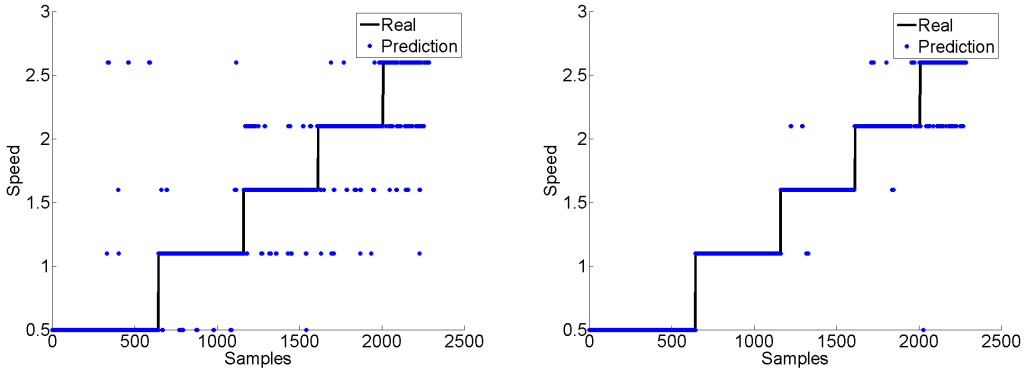
Speed prediction should determine which walking or running speed is intended by the user. For each of the two gaits a different set of speeds is considered. In case of walking, the set of speeds is constituted by 0.5, 1.1, 1.6, 2.1 ans 2.6 m/s. The slowest speed of this set is 0.5 m/s and might be chosen by elderly people. To cover a broad range of speeds, the speeds increase until 2.6 m/s which is about $0.6m/s$ above the preferred transition speed from walking to

running [43]. Also for running a broad range of speeds is covered. All running speeds are given in Table 3.1.

Speed Classification

Since both gaits consider a countable amount of speeds, multi-class classification can be applied. Therefore, binary classification is achieved with support vector machines and extended to multi-class classification according to Section 2.2.2. In case of walking there are 5 and in case of running 7 different speeds or rather classes. For each class, the respective speed serves as label. As input we use different input signals and different τ in context of a window-in-time approach. Changing the used input or τ leads to different classification results. For instance, the comparison of different input signal combinations reveals which inputs give relevant information for inferring the intended speed. Increasing τ may also give additional information to the classifier and can improve classification performance. In the following, the impact of both variable parameters is analyzed. At first, τ is fixed to one.

Classifiers with single signals as input, e.g. angle only, give poor performance. The performance increases when combining different sensors, but the resulting classifiers are still not sufficient good. In more detail, the accuracy of the classifiers is mostly below 80%. A higher accuracy (86%) is only achieved when using a classifier based on the combination of all sensors (angle-vel-Ax-Ay-fzl). The performance of this specific classifier is shown in Figure 3.9a. But as defined in Section 3.2, the accuracies should be even higher for the purpose of active ankle control. This is especially important for the conducted experiments, since the differences between the considered speeds are noticeable. If the used classifier is wrong, the user will be forced to an unintended speed which might be uncomfortable or even cause the user to stumble. Other important factors are how long the speed is misclassified and if the wrongly predicted speed is an adjacent class of the real speed or farer away. The impact to the user increases with the misclassification period and the speed difference. Additional problems can occur if the classifier constantly switches between prediction results even though the intended speed is constant. This problem is known as chattering and can be avoided by a voting scheme as in [20]. A voting scheme of length l stores the last l predictions and determines the speed according to a majority voting. In this case, a speed classification is only applied, if



(a) $\tau = 1$ **(b)** $\tau = 5$

Figure 3.9.: Speed prediction for walking. Both predictors are realized as support vector machines and use the combination of all sensor signals as input. The left panel is for $\tau = 1$ and the right panel for $\tau = 5$. Notice that the overall performance increases with the window size.

Table 3.6.: Varying widow size for speed prediction based on support vector machines and a window-in-time input-output setup. Each cell gives the accuracy in percent. The abbreviations for the input signals used in the column headings are given in Table 3.2.

τ	angle	vel	Ax	Ay	fzl
1	35	30	35	34	36
5	62	59	50	47	50
10	80	71	58	58	62

τ	angle-vel	Ax-Ay	angle-vel-fzl	Ax-Ay-fzl	angle-vel-Ax-Ay	angle-vel-Ax-Ay-fzl
1	66	46	79	63	77	87
5	80	65	90	83	89	94
10	87	75	93	93	91	97

a certain percentage of the last l prediction results is in accordance. The voting scheme helps to avoid chattering and short periods of misclassification, but it also introduces some delay. For instance, if a sudden speed change occurs, it takes some samples until the percentage needed for accordingly classifying the new speed is reached. Note that the larger l, the larger the delay. Since switching between speeds should be as fast as possible, l should be either small or a voting scheme should be avoided. Ideally, also continuous transition should be recognizable.

Regarding Figure 3.9a again, two things are noticeable. First, misclassification results are only a class below or above the original one and not farer away. Second, there are areas where more than 5 consecutive samples are classified wrong. This can only be compensated with a large voting scheme which implies a large delay. Instead of using a large l or a voting scheme at all, it might be possible to avoid such consecutive misclassification by increasing τ.

Even though the single input classifiers improve when increasing τ to 5, they still give an accuracy below 80%. Also increases to $\tau = 10$ does not improve applicability. The best single input is shank angle with an accuracy of 80%, followed by shank velocity. Force plate or both of the accelerometer signals on their own are less important for distinguishing speeds and consequently give a smaller accuracy. All obtained results for increasing the window size are given in Table 3.6.

Regarding $\tau = 5$, shank or accelerometer data alone is still not sufficient. Only when combining those signals with each other or with force plate data, acceptable performance is achieved (above 86% accuracy). The predictor for shank angle, shank velocity and the acceleration in x- and y-direction, for instance, achieves an accuracy of 89%. Force plate data as additional input boosts the performance of the considered example to an accuracy of 94%. Although force plate data is not useful as single input, it generally boosts the classification performance when used as additional input. The reason for force data not being useful on its own, is that during swing phase no force is given and consequently all speeds look the same. The classifier with 94% accuracy (angle-vel-Ax-Ay-fzl) is the best performing one for $\tau = 5$. It's prediction results are shown in Figure 3.9b. As for $\tau = 1$, the prediction results contain some consecutive samples that are classified wrong. Moreover, the predictions for walking at $2.6m/s$ are often wrong. This could be due to the test subject's problems with this specific walking speed (see Figure 3.14). For the subject it was hard to keep pace, since the speed is above the subject's normal walking speed. This can be seen when regarding the magnitude

of the force measurements. The magnitude increases, but normally is should stay the same for walking at constant speeds. So, the test subject might have applied some pattern which is more similar to a slower walking speed or which might include some elements characteristic for running. Also note that the speed of $2.6m/s$ is above the preferred transition speed from walking to running.

When increasing τ to 10, the performance of the classifiers increases as well. All classifiers that use input signal combinations achieve an accuracy close to 90% or above 90%. The performance differences between different input combinations is as for smaller window sizes. So, shank data or accelerometer data alone is compared to other combinations not worth using. When combining both, better results are achieved. The usage of force data in addition still leads to the best classifiers. Consequently, the classifier for shank data, accelerometer data and force data (angle-vel-Ax-Ay-fzl) is the best one. This classifier gives an accuracy of 96% which is appropriate for the context of active ankle control. A smaller window size, however, would be more favorable. For $\tau = 10$ it might take some time until fast speed changes are recognized, since all old sensor measurements need to be flushed. Also a force sensor is required, but the current prosthesis is not equipped with such a sensor. So, an equally performing approach that does not require a force sensor would be a plus.

Note that the classification performance depends on the considered speed setup. If the considered speeds are more close, the classification error might increase because the input data might be less distinguishable. Furthermore, all classifiers are based on the multi-class classification techniques presented in Section 2.2.2. For the results given above the one-versus-one approach was applied. In case of the one-versus-all approach the results are nearly identic.

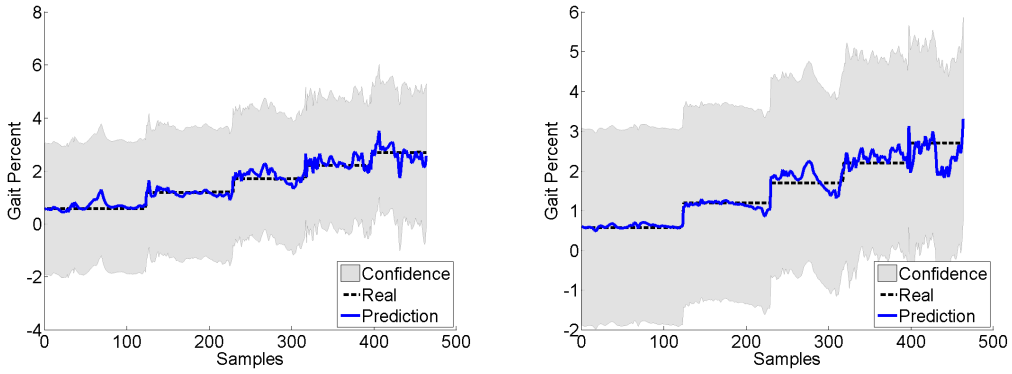
Regression

As stated earlier, we aim at predicting the speed intended by the user. For each gait (walking/running) a fixed amount of speeds is possible or rather predictable. The predictor should also perform for fast transitions between speeds. This means that the speed the user changed to is recognized as fast as possible. A small window size is consequently beneficial. Ideally, also continuous transitions between speeds should be identifiable. If the speed, for instance, increases slowly from $0.5m/s$ to $1.1m/s$, it would be desirable if the predictor or classifier notices this as well. One approach is to use the probability add on for the support vector ma-

Table 3.7.: Varying widow size for speed prediction based on Gaussian process regression and a window-in-time input-output setup. Each cell gives the prediction error(+/-) in m/s. For the input signal abbreviations used in each column heading, see Table 3.2.

τ	angle	vel	Ax	Ay	fzl
1	1.0/1.5	1.3/1.6	1.5/1.5	1.5/2	1.3/1.5
5	0.8/1.0	1.0/1.7	1.0/1.5	1.5/1.5	1.2/1.6
10	0.7/0.6	1.0/1.7	1.5/1.0	1.0/1.2	1.2/1.6

τ	angle-vel	Ax-Ay	angle-vel-fzl	Ax-Ay-fzl	angle-vel-Ax-Ay	angle-vel-Ax-Ay-fzl
1	0.9/1.0	1.5/2.0	0.8/0.6	1.5/1.5	0.7/0.7	0.6/0.7
5	0.6/0.6	1.3/1.5	0.7/0.5	0.6/0.6	0.5/0.5	0.5/0.6
10	0.5/0.6	1.0/1.0	0.6/0.5	0.5/0.6	0.4/0.5	0.6/0.5



(a) $\tau = 1$ **(b)** $\tau = 10$

Figure 3.10.: (a) Speed prediction results for walking based on Gaussian process regression. As input the combination of shank, accelerometer and force data is used. The window size is set to one. (b) The same predictor as in (a) with τ increased to 10. When increasing τ, the prediction performance improves mostly for slower speeds.

chines, to interpolation between speeds (compare Section 2.2.1). Another approach which naturally incorporates continuous outputs is the Gaussian process framework. This approach might also give speed prediction errors that are smaller than for the classification case, since the predicted speed can be in between two classes. As for classification, a speed predictor is learned for each gait. The given problem, however, is rather a classification problem than a regression problem. For example, in case of walking only five different outputs are available and are used for training. Even though the setup is not optimal, it is compared with respect to classification. Therefore, we exemplary use the speed predictor for walking with a window-in-time setup and different τ. The obtained prediction results are shown in Table 3.7.

For $\tau = 1$, the single input predictors are not usable at all. Their predictions can give errors about $1.5m/s$ and look more random than based on input information. To evaluate prediction errors, the error can be compared to the errors occurring for classification. If an input is misclassified, it is mostly classified as speed above or below the real one. The distance between such adjacent speeds is about $0.4m/s$ and consequently the classification error is mostly $0.4m/s$ as well. Combinations of input signals give more information to the predictor and improve prediction performance to some degree. The relationship between the prediction performance of the different input signal combinations is the same as for classification. So, both machine learning techniques identify the same input signals as relevant for speed prediction. Hence, combining shank data with additional sensors gives compared to other combinations good performance. So, the best predictor for $\tau = 1$ is still achieved when combining all sensors (vel-angle-Ax-Ay-fzl). This predictor is, apart from the last speed, within an error of about $\pm 0.6m/s$. For the last speeds all classifiers show some larger errors like $\pm 0.8m/s$. As described for classification, this might be due to the problems of the test subject with this speed. Furthermore, Figure 3.10a reveals that the prediction results are varying a lot. In most situations, there are a lot of ups and downs so that the speed above or below is more likely. All in all, the predictor is not accurate enough for our purpose. When increasing τ the single signal predictors still remain unusable, the predictors for Ax-Ay-fzl and vel-angle-Ax-Ay-fzl do not change much with respect to the error and the other predictors decrease in

Table 3.8.: Prediction time [s] for on gait cylce

τ	**SVM**	**GP**
1	0.005	0.053
5	0.007	0.061
10	0.009	0.063

prediction error. Along with the prediction error, also the underlying function changes. For vel-angle-Ax-Ay-fzl for example, the prediction results for $\tau = 1$ (Figure 3.10a) and $\tau = 10$ (Figure 3.10b) look very different although the error does not change much. With increasing τ the prediction results for the slower speeds get more narrow to the real ones. For faster speeds, however, the function still varies up and down.

Comparison

For the purpose of speed prediction with fixed classes, classification is, as expected, more promising. The Gaussian process setup can incorporate continuity, but gives speed prediction errors of the same or bigger size. Moreover, it is in the most cases only close to the real speed, whereas classification is right. However, we must consider that the Gaussian process is trained only with 5 different speeds as output values. Also note that slower transitions between speeds are not analyzed here. Section 4.3, in contrast, will inspect a special kind of transitions as well. Besides prediction performance, we compare also prediction time. As in case of gait percent prediction, the time to predict the speed for one gait cycle is evaluated. The usage of more input signals for a fixed τ influences prediction time just marginally, So, Table 3.8.

compares prediction time only for different τ. Since their solution is sparse, support vector machines are in general faster than Gaussian process regression. Predictions for new inputs consequently require only a small number of training points. Gaussian process regression, in contrast, requires to evaluate the kernel function for combinations of the input with all training points.

Limited Predictors

The support vector machine and the Gaussian process approach need both a relatively large window sizes to give acceptable results. However, we want to achieve smaller window sizes to facilitate a more independent controller that can detect abrupt changes as fast as possible and needs less time for initialization. Furthermore, the best classifiers require to equip the prosthesis with an additional sensor for force data. To avoid the additional sensor, an equally performing approach which uses only shank and accelerometer data should be found. Some difficulties of the presented approaches result from the relatively large area it should perform predictions for. Both approaches must differ between speeds for all possible values of gait percent. Instead it might be easier to distinguish between speeds when focusing on a special point during the gait cycle. Therefore, the gait cycle is divided into prediction areas. The prediction areas depend on gait percent and cover the whole gait cycle. For each area a unique speed predictor is learned and used. A very simple example is to divide the gait cycle into two areas, where one area covers 1–50 gait percent and the other one lasts from 51 to 100 gait percent. Consequently, speed prediction depends on gait percent prediction. The gait percent value is provided by the gait percent predictor and is used to chose the right area or

Table 3.9.: Comparison of the classification accuracies achieved by two limited predictors. The gray shaded values are the accuracies for a predictor limited to $10-20$ gait percent, whereas the other values are for a predictor for 60–70 gait percent. Note that the used input signal abbreviations are given in Table 3.2.

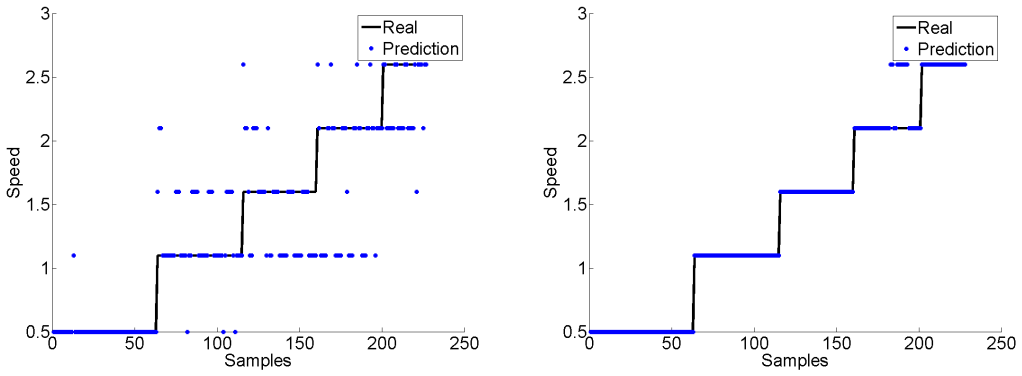
τ	angle		vel		Ax		Ay		fzl	
1	41	77	40	70	55	50	60	70	59	38
5	66	94	63	85	69	93	60	85	77	54
10	88	96	76	93	88	99	67	89	84	59

τ	angle-vel		Ax-Ay		angle-vel-fzl		Ax-Ay-fzl		angle-vel-Ax-Ay		angle-vel-Ax-Ay-fzl	
1	63	96	84	96	74	95	74	96	85	100	84	100
5	88	99	96	99	88	98	93	99	94	100	96	100
10	96	100	98	99	96	100	99	100	96	100	98	100

rather speed predictor. Because gait percent prediction incorporates an error of about ±5%, a realistic size for each area is a span of about 10 gait percent. So even if the gait prediction is wrong by 5%, the right area is chosen. A more sophisticated approach is to define a separate area for each gait percent value between 1 and 100 (value ±5%) and to evaluate only the predictors for those areas which are within the gait percent prediction error. The final speed is determined by a majority voting. In the following it is investigated if speeds are easier to discriminate when using predictors restricted to specific areas. Here, we exemplary evaluate the predictors for the regions of 10–20 and 60–70 gait percent. As can be seen later, both areas together are sufficient to categorize the prediction behavior for the whole gait cycle.

For both gait percent areas a support vector machine with a window-in-time setup is used. As before, we evaluate the prediction results for τ equals 1,5 and 10. At first, the predictor for 10–20 gait percent is compared to the predictor for the whole gait cycle. Overall, the prediction performance of both reveals the same relations between the performance of different input signal combinations. Also the prediction performance increases for both with increasing window size. However, for $\tau = 1$ the accuracy of the predictor limited to 10–20 gait percent is about 4–10% higher as for the unlimited classifier. Only in case of angle-vel-Ax-Ay-fzl and $\tau = 1$ the accuracy does not improve. All in all, the predictors for $\tau = 1$ are still not accurate enough for real world applicability. When increasing τ, the accuracy remains still about 4–10% above the one for the same setup in the unlimited case. So, for $\tau = 5$ there are already classifiers with an accuracy above 90% which do not require force data as input. Moreover, several classifiers reach an accuracy close to 100% when increasing τ to 10. The comparison reveals that the restriction of speed prediction to an unique gait percent area boosts classification performance. For the classifier for 60–70 gait percent, the performance is boosted even more. Before details are revealed, note that the relation between the performance of different input signal combinations is also for this classifier equal to the relations concerning the unlimited case. The predictors performs so good that even some classifiers for $\tau = 1$ achieve an accuracy close to 100%. An example is the classifier for shank angle and velocity as input which gives an accuracy of 96%. All prediction results can be seen in Table 3.9

When increasing τ, the classifiers with an accuracy close to 100% only increase a little, at which some classifiers even reach 100%. For the larger window sizes, also single input



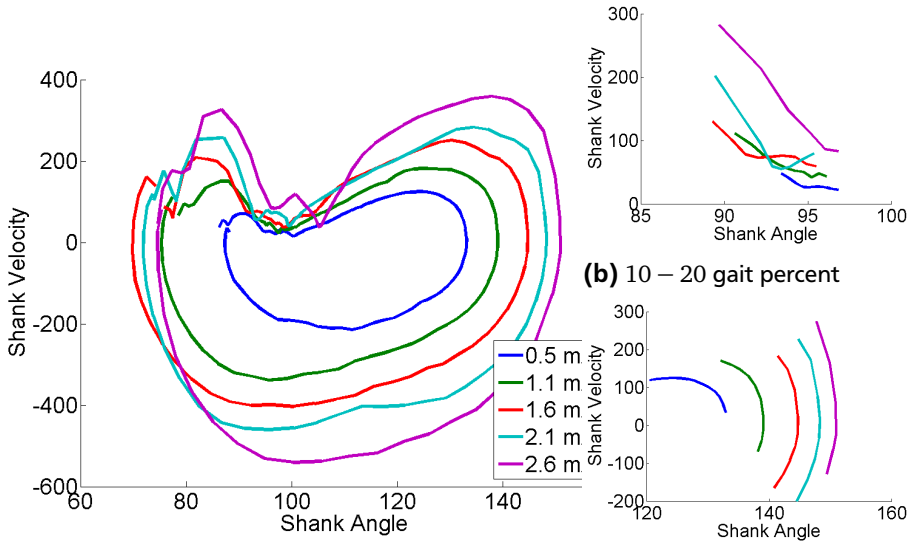
(a) Predictor limited to $10 - 20$ gait percent **(b)** Predictor limited to $60 - 70$ gait percent

Figure 3.11.: Different limited speed predictors or rather classifiers with $\tau = 1$ and all sensor data as input. (a) shows the limited predictor for the area $10 - 20$ gait percent, and (b) is for the area $60 - 70$ gait percent. The second predictor achieves a higher accuracy, since its input data is easier to distinguish.

predictors, e.g., the classifier with only Ay as input, can improve to applicability. An important question to ask is: Why is the performance that good and what constitutes the differences between the different areas? The question can be answered by having a look at the input signals itself. Here, the classifiers for the input signals of shank velocity and shank angle are examined. For $\tau = 1$, the classifier for the area of 10–20 gait percent gives an accuracy of 63% and the classifier for 60–70 gait percent achieves 96% accuracy. Figure 3.11a compares the prediction results graphically.

The input signals of shank angle and velocity are depicted in Figure 3.12.

Panel 3.12a gives the input signals for the whole signal, whereas panel 3.12b and 3.12c show the specific areas considered here. In all panels, the shank angle is plotted against the shank velocity. The overall figure, which is also known as angle-velocity cyclogram, shows that the distance between origin and angle-velocity curve increases with speed. However, there are gait percent areas where the curves for different speeds get close or intersect each other. For instance, for 10–20 gait percent the input values for different speeds are so close that it is hard to discriminate between speeds given only one measurement. That is why more measurements are needed to achieve sufficient classification results. In case of $60 - 70$ gait percent, the speeds are clearly separated so that they can be easily distinguished graphically and consequently also with a support vector machine with window size 1. Regarding the angle-velocity cyclogram again, it is evident that the whole cyclogram can be described with the use of the two analyzed areas. On the one hand, there are areas where the input signals get close and the classifier will perform as for the area of 10–20 gait percent. On the other hand, many areas show also obvious differences between speeds as observed for the area of 60–70 gait percent. So, the overall classification performance for each area can be appreciated by just evaluating the performance of both areas regarded here. All in all, the performance increases by dividing gait percent in different areas and using an unique speed classifier for each one. For areas where the inputs are close or overlapping, the window size needs still to be larger. In the other case, however, window sizes of one can be used. Furthermore, it is no longer required to rely on force data to get prediction results acceptable for active ankle control. One reason why force data improves the prediction results for the unlimited case is that all other inputs are close near heel strike. Since the force values for the impact



(a) Whole gait cycle

(b) $10-20$ gait percent

(c) $60-70$ gait percent

Figure 3.12.: Shank angle plotted against shank velocity for different speeds. (a) shows shank angle and velocity for the whole gait cylce. (b) is limited to the area of $10-20$ gait percent, whereas (c) is limited to $60-70$ gait percent. Using the depicted data as input, different speeds are for the first area much harder to distinguish than for the second area.

at heel strike increase for faster speeds, they can be used to improve discrimination. The improvements achieved by using several predictors introduce also an additional dependency. The speed prediction relies on gait percent prediction and might be confused if gait percent prediction fails.

For classification, acceptable results were achieved by restricting the classifier to a specific gait percent area. The same experiments are also performed for Gaussian process regression. As for classification, the predictors for both considered areas improve the prediction performance compared to the unlimited case. In more detail, some predictors for 60–70 gait percent achieved already for $\tau = 1$ prediction errors within $\pm 0.3 m/s$. In case of 10–70 gait percent, the window size needs to be increased to achieve such results. When increasing the window size for the predictors for 60–70 gait percent some input combinations can limit the error even to $\pm 0.2 m/s$. So for the Gaussian process framework, the prediction performance does also depend on how close or overlapping the input data is.

3.5 Gait Prediction

Given sensor data obtained by the prosthesis, gait prediction discriminates between all possible gaits. In this thesis the gaits walking and running are considered. So, we have a binary classification problem that can be solved with the use of a support vector machine. If more than two gaits are considered, e.g., walking, running and standing, support vector machines are still applicable. In this case, binary classification is extended to multi-class classification as described in Section 2.2.2. To evaluate gait prediction with support vector machines, a window-in-time approach is chosen and the obtained results for different window sizes and signals combinations are compared. For the input signals we use the database generated from the motion capturing recordings. The classification output is either walking or running. In the

Table 3.10.: Gait prediction results when using a support vector machine with a window-in-time setup. Each cell gives the accuracy of its corresponding predictor and window size in percent. The input signal abbreviations used in the column headings are introduced in Table 3.2.

τ	angle	vel	Ax	Ay	fzl
1	68	56	57	61	71
5	71	64	58	69	73
10	87	79	73	80	76

τ	angle-vel	Ax-Ay	angle-vel-fzl	Ax-Ay-fzl	angle-vel-Ax-Ay	angle-vel-Ax-Ay-fzl
1	72	67	86	77	94	96
5	89	87	94	93	96	98
10	94	94	97	96	97	99

following running is decoded as 1 and walking as 0. The prediction performance is evaluated for the window size of 1,5 and 10. All obtained results are shown in Table 3.10.

As for gait percent and speed prediction, prediction performance increases with window size. Furthermore, the results for single input predictors are also for gait prediction not worth using. In case of $\tau = 10$, the best single input predictor achieves an accuracy close to 90%. An useful gait prediction, in contrast, should ideally achieve an accuracy more close to 100%. The reason therefore is twofold. One the one hand, high accuracies are required for safety reasons. On the other hand, gait percent and speed prediction rely on gait prediction and might be fooled when the gait mode is misclassified. When combining input signals, shank data is slightly superior to accelerometer data, e.g., for $\tau = 1$ angle-vel achieves 72% and Ax-Ay 67% accuracy. The combination of both (angle-vel-ax-ay) increases performance further and is with 94% accuracy ($\tau = 1$) already applicable in context of active ankle control. An

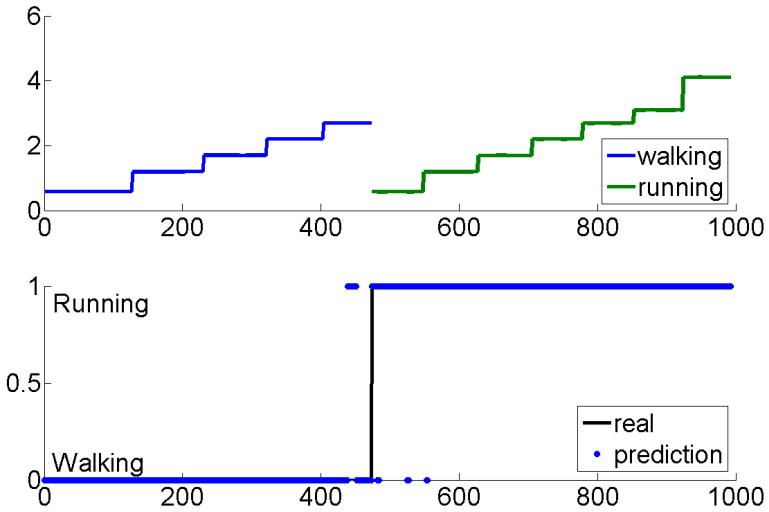


Figure 3.13.: Gait prediction based on a support vector machine with $\tau = 2$ and shank and accelerometer data as input. The figure above display the input and the figure below the corresponding prediction result. Note that the most prediction errors happen for walking at $2.6m/s$. That might be due to the test subject's problems with this walking speed.

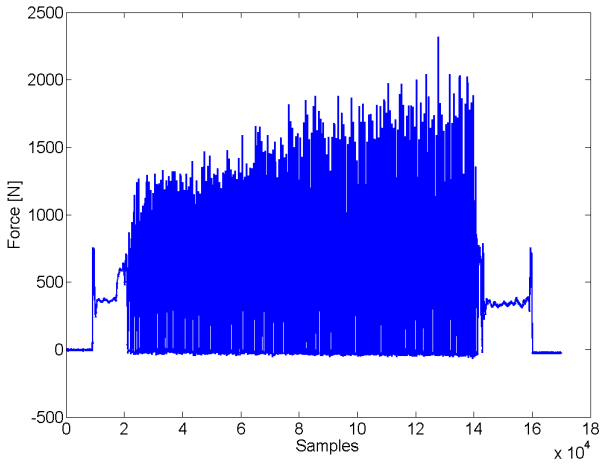


Figure 3.14.: Force data triggered by the right foot during walking at $2.6m/s$. The test subject has problems with keeping pace and therefore continuously increases force. For constant walking speeds, however, the required force should normally be similar.

higher accuracy is only achieved when adding force data to the considered classifier. The resulting classifier gives 96% accuracy for $\tau = 1$ and 99% for $\tau = 10$. This classifier, however, requires to mount an additional force sensor to the prosthesis. So, we can decide if the accuracy of the classifier for shank and accelerometer data as input is satisfactory or if the improvements due to the force sensor are significant enough to add an additional sensor to the prosthesis. Moreover, the window size introduces a trade-off between speed and accuracy. The predictors for smaller window sizes are faster, but a not that accurate. For instance, the predictor for angle-vel-ax-ay gives 94% accuracy for $\tau = 1$ and 96% accuracy for $\tau = 5$. Even though the differences in accuracy are not that big, the window size can be fine tuned. In case of the given example, it can be investigated which window size is needed to achieve an accuracy in between those for $\tau = 1$ (94%) and $\tau = 5$ (96%). It turns out that already $\tau = 2$ gives an accuracy of 95%. The corresponding classification results are shown in Figure 3.13.

Note that the area for walking at $2.6m/s$ shows the most errors. As in Section 3.4, this is due to the test subject's problems with this walking speed. The problems become visible when regarding the corresponding force measurements given in Figure 3.14. The forces induced by both feet increase continuously. Moreover, $2.6m/s$ is above the preferred transition speed from walking to running. Both facts might indicate that the subject already produced a patter similar to running.

3.6 Supervisory Controller

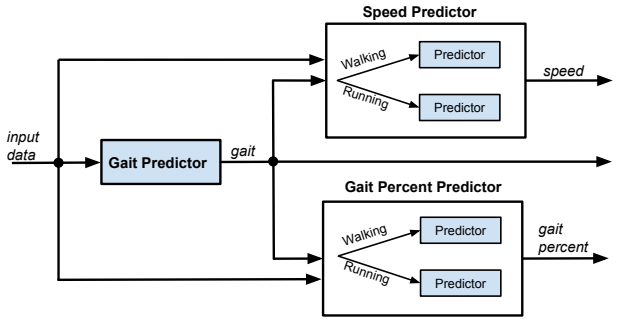
The supervisory controller consists of components for gait, speed and gait percent prediction. Previously, each of this components was designed and evaluated in isolation. The obtained results are shortly summarized, to give the most applicable setup for each component. Afterwards, we purpose how to construct the supervisory controller given all components. Note that all results are based on motion capturing data. Since motion capturing data is compared to sensor data really accurate and noise free, it is perfect for evaluating feasibility.

For gait percent prediction, Gaussian process regression is used. The predictor outputs the phase of ankle movement within a gait cycle which is in between 0 and 100 gait percent.

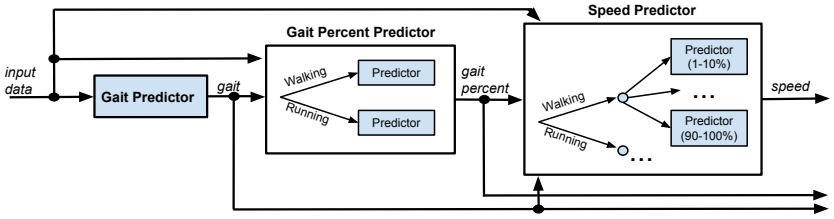
At heel strike gait percent is reseted to 0% and subsequently it increases until 100% shortly before the next heel strike of the same leg. Either gait percent predictors for each gait-speed combination or a single gait percent predictor for each gait can be considered. The second approach is sufficient, since it was shown that gait percent prediction generalizes along speeds. Consequently, gait percent prediction depends only on gait prediction. Further evaluations revealed the window-in-time setup with angle-vel as input and $\tau = 1$ as most applicable setup. Both setup parameters, namely the used input signal combination and the window size, are optimal. Regarding the input signal combination, only measurements obtainable with a gyroscope are required. This sensor setup is sparse and already available, because a gyroscope is mounted to the considered prosthesis. In case of window size, the smallest possible size is achieved. The smaller the window size, the faster abrupt changes are recognized and corrected for. Furthermore, the error of the considered predictor is always within ± 5 gait percent. At first, the error was higher for transitions between two gait cycles. This is due to the large step from 100 to 1 gait percent and is corrected for by transforming the predictor's output. For more details see Section 3.3. Speed prediction is in contrast to gait percent prediction more problematic. Since a fixed amount of walking and running speeds are considered, we face a multi-class classification problem. The problem is solved by using a classifier based on support vector machines for each gait. The results give only good performance when using large window sizes and as many input signals as possible. In terms of active ankle control just two predictors give applicable performance. The first predictor is the one for angle-vel-ax-ay as input and $\tau = 10$. This classifier improves a little further by using force measurements as additional input, yielding the second predictor. Even though the prosthesis is not equipped with a force sensor, the prediction results can be used for an informed trade-off. The window size can be reduced by dividing the gait cycle in several areas and learning a predictor for each of those. For instance, there could be predictors for $1 - 10$ gait percent, $10 - 20$ gait percent and so on. In this case, gait percent prediction is used to identify the corresponding speed predictor. The window size required by each predictor does also depend on the gait percent area. For many areas, the input signals are easy to distinguish and we can use a small window size. For some other areas, however, the input data is more close and we need large window sizes again. Gait prediction discriminates between gaits such as walking and running. Therefore, a support vector machine with a window-in-time setup is used. As for speed prediction, the best performance is achieved when using as many input signals as possible. So, the best predictor is constituted by angle-vel-ax-ay-fzl as input. The next best predictor is given by removing force data from the inputs. This predictor achieves on average an accuracy reduced by 2%. So, the question of adding a force sensor or not becomes apparent again. In contrast to speed prediction, gait prediction performs already good for small window size like one or two.

Regarding the results for gait, speed and gait percent prediction, two supervisory control structures are possible. Both structures differ just by means of speed prediction and are displayed in Figure 3.15.

The supervisory controller is responsible to determine gait, speed and gait percent. Therefore it proceeds as follows: In a first step, gait is estimated. When gait is known, the gait percent prediction can take place. For speed prediction two possibilities are imaginable. The first approach is based on a single speed predictor for each gait. Consequently, a larger window size is required, but the predictor does not depend on gait percent prediction. The second approach uses several speed predictors, where each is specific for a gait percent area. For the



(a) Supervisory controller with gait percent independent speed prediction.



(b) Supervisory controller with gait percent dependent speed prediction.

Figure 3.15.: Two possible supervisory controllers. For both controller, gait prediction is the fist step. When gait is known, gait percent prediction can take place. In Panel (a) speed prediction is also possible as soon as gait is known. For Panel (b) speed prediction depends additionally on gait percent prediction. Therefore smaller window sizes can be achieved to some extend.

most gait percent areas, smaller window sizes are possible. However, not all gait percent areas give decreased window sizes and we get the dependency to gait percent prediction in addition.

4 Sensor Data Based Evaluation

Chapter 3 revealed that supervised machine learning is useful for predicting the user's intent during walking and running. In this case, the user's intent consists of the desired gait, speed and gait percent. Gait percent, also known as gait phase, describes the current state of the ankle movement. At heel strike or rather at the beginning of a gait cycle, gait percent is 0%. During gait cycle, it increases continuously until it reaches 100% shortly before the next heel strike of the same leg. Predicting gait, speed and gait percent are three separate tasks, but all together they form a supervisory controller for powered ankle control. When knowing all three outputs, the actual control signal or rather the current for the prosthesis's motor can be determined by a simple lookup. This control signal is enforced with a PD controller. In the previous section it was shown that gait percent and gait prediction perform good. For gait percent prediction, shank angle and velocity are good predictors. In case of gait, gyroscope and accelerometer data are used. Moreover, gait percent and gait predictors perform already for window sizes of $\tau = 1$. Speed prediction, in contrast, requires some tweaks to be as accurate as desirable. Either a large window size or several limited predictors are required. Both predictors rely on gyroscope and accelerometer data as input. When using force data in addition, performance increases a little.

In Chapter 3, the inputs for the predictors were computed from data recorded by a motion capturing system. If compared to real sensor data, motion capturing data is really accurate and noise free. This section aims at demonstrating the real world applicability of the supervisory controller based on machine learning. Therefore, the supervisory controller must be wearable and, thus, it must be able to operate with the sensor data provided by the prosthesis. The prosthesis operates with an inertial measurement unit. More details about the sensor and what data is available is given in Section 4.1. The two following sections show that gait percent and speed prediction also work in case of sensor data. The results for sensor and motion capturing data are similar, so that motion capturing data can be seen as good simulator for control mechanism based on machine learning. Note that gait prediction is not examined for the sensory data. Instead, we focused an on the improvement of speed prediction. As result we introduce a feature that dramatically improves speed prediction. In Section 4.4 we conclude with a summary of sensor based prediction results and purpose an accurate and wearable supervisory controller.

4.1 Database Generation

To evaluate the machine learning techniques used for the supervisory controller in Chapter 3 also for the real prosthesis, a database for the prosthesis's sensory data is created. This database is used to generate distinct training and test sets for the tasks of gait and speed prediction. A first part of the evaluation is about performing the same experiments as in Section 3 with the real sensor data as input. While doing so, different predictors constituted by different setups and combinations of inputs signals are learned and compared with the corresponding predictors for motion capturing data. So,

it is tested if the noisier sensor signals can also constitute as accurate predictors as for motion capturing data. The comparison, however, is only done for walking and not for running. Consequently, gait prediction is not considered here. Instead, more close walking speeds are used for the experiments. For an overview of all speeds see Table 4.1. This decision was made due to the fact that gait percent and gait prediction perform as accurate as required, but speed prediction should be improved. By improving speed prediction, the proposed controller will gain the most. So, the second task of this section is the improvement of speed prediction. For achieving and proofing this, we wanted to have more closely located speed measurements. We assume that closer speeds will complicate the proposed speed prediction mechanisms, since the input data will be less distinguishable. If our assumption is true, some different methods for speed prediction must be found. Moreover the different speed measurements can be used to test if gait percent prediction also generalizes to more detailed speeds.

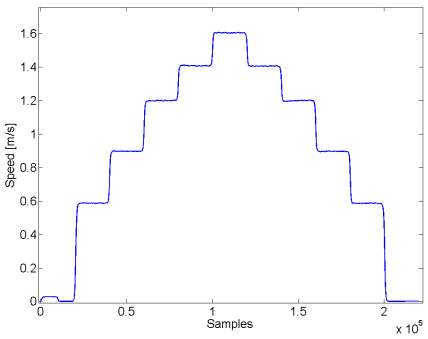
Table 4.1.: Considered speeds

Walking Speeds [m/s]
0.5, 0.8, 1.1, 1.4, 1.6

The recordings for the database were performed with the same test subject and the same treadmill. As reminder, the subject was a healthy male person of 1.86 m height and 76 kg weight. The treadmill has two force plates, each of the two measures the impact of one foot (left and right) on the treadmill. Instead of the motion capturing system, an inertial measurement unit attached to the shank of the subject was used to record the sensor signals. This inertial measurement unit consists of a biaxial gyroscope and accelerometer. The used gyroscope is the ST LPY550AL. As accelerometer the MMA7361L of Freesacle Semiconductros is used. The sensor was attached at the shank, more in detail about 20 cm above the ankle. At this place the sensor of the active ankle prosthesis is located as well. For the detailed mounting, see Figure 4.1a,

The experiments performed in Chapter 3 require signals such as shank angle, shank velocity and acceleration data in x and y direction. Shank velocity is provided by the gyroscope and acceleration data is given by the accelerometer. The shank angle, in contrast, is not given directly. However, it might be derived from shank velocity instead. An obvious approach for computing shank angle is the numerical integration of shank velocity. Another possible approach is based on a transfer function and is described in [11]. The derivation method

(a) Sensor attachment.



(b) recording procedure.

Figure 4.1.: (a) sensor setup for database generation (b) recording procedure. All speeds are walked for a given time. At first speeds are increased until the maximal speed is reached. Afterwards speeds are decreased until the treadmill stops.

must be the best approach for our context, which is the prediction of gait, speed and gait percent. That is why the approach for deriving shank angle cannot be chosen in isolation. The next section uses gait percent prediction to describe in detail which approach is the best one to go for.

The force plates provide with their force measurements a signal which cannot be computed by the inertial measurement unit. This data is kept anyway, because it is used to perform some of experiments of Chapter 3 . In Chapter 3 it was moreover shown that force data can slightly improve prediction performance. Consequently, it can be analyzed if it is advantageous to mount a force sensor in addition. In Section 3.1, the motion capturing and the force plate data were automatically aligned, since both were controlled and recorded by the same computer. For the sensory data recordings, the inertial measurement data and the force data is not aligned. That is because the sensor data and the force plate data are recorded by different computers. The force plate data is again given by the computer that controls the treadmill and the sensor data is recorded by the computer that also controls the real prosthesis. After performing the recordings, the data is aligned by finding the first heel strike in the force and the sensor measurements.

For the motion capturing data, a separate data recording was performed for each gait-speed combination. Here, all speeds are recorded after each other. The full recording procedure is given in Figure 4.1b Summarized, each speed given in Table 4.1. is walked for 20 seconds. After finishing one speed, the treadmill directly increases the speed to the next considered one. Note that the speeds in Table 4.1 and in Figure 4.1b differ by $0.1m/s$. Even though the treadmill was adjusted to the speeds given in Table 4.1, the produced speed is different. Of course the treadmill cannot directly jump to the next speed, resulting in short transitions shaped like a power function of degree 3. Also humans cannot change the speed directly and consequently realistic transitions are obtained. So it is also possible to evaluate the transition behavior of the learned predictors. For gait percent prediction it can be evaluated if it also performs for speed transition. In case of speed prediction, it is interesting to see if the transition itself are noticed. After reaching the maximal considered speed, all speeds are walked in decreasing order as well. So, increasing and decreasing transitions are available for testing.

Note that in the following the same input signal abbreviations as introduced in Section 3.1 may be used. Also the performance objectives of Section 3.2 are still valid.

4.2 Gait Percent Prediction

Before performing the experiments conducted in Section 3.3 with the sensory data as input, it is required to determine the best way for deriving shank angle. This can either be done by integrating the shank velocity provided by the gyroscope or by applying a transfer function to it. Normally, integration is the most common approach to numerically determine the angle or position given the corresponding velocity. But the numerical integration of the shank velocity leads to an integration drift. The drift is too heavy for obtaining an useful signal. Consequently, gait percent prediction where shank angle is one of the input signals is not applicable. To avoid the drift, it is required to reset the integration bounds after some time. A common approach, e.g., for walking speed estimation with inertial measurement units [24], is to reset the integration after each stride cycle. A recognizable event for resetting the integration is the heel strike. By using heel strike as reset point, the integration drift is

avoided and the resulting shank angle is in shape and dimension similar to the shank angle computed from the motion capturing system. The prediction performance for specific gait-speed combinations with shank angle as one of the input signals is similar to those when using the motion capturing data. Even though the obtained shank angle is similar to the one computed from the motion capturing system, the data does not generalize to all speeds. For instance, an untransformed predictor for all walking speeds with shank angle and velocity as input and $\tau = 1$ gives no good prediction performance (error of $\pm 14\%$). This may be due to the reset of integration bounds, which implies that the initial shank angle for a stride cycle is for all speeds alike. So, the data is somehow misaligned and the generalization ability for different speeds is lost. Instead, more sophisticated integration approaches or the simple transfer function approach given by Holgate et al. [11], which is considered here, can be used. For their tibia based control, Holgate et al. uses a transfer function to obtain the tibia angle from the tibia angular velocity. The transfer function is given by

$$\frac{\tau^2 s}{(\tau s + 1)^2}, \tag{4.1}$$

where τ is a non-negative, free parameter [11]. The given transfer function is stable and τ influences how fast the transfer function is. Because of stability, initial conditions can be arbitrary and the transfer function can be applied without resetting bounds. The faster the transfer function, the faster the integration drift is compensated. However, τ cannot be choosen arbitrary large. If τ is too larger or rather the transfer fuction too fast, the characteristic tibia angle shape is lost [11]. Holgate et al. give another beneficial property of the transfer function, namely the deindivudualization of tibia angle between unique subjects. The transfer function deindivudualizes, since it gives not the real tibia angle but a value that is similar in shape and centered around zero. It turns out that the given transfer function is also sufficient for our use case. Of course, the result is of not the real shank angle, but its similar in shape and centered around 0. Even though it gives not the exact value, it is useful, because it provides generalization for all speeds. From a machine learning point of view, the exact interpretation of the input signal is not important as well. It is enough if the input value facilitates good predictions.

Given shank angle, all input values needed for the experiments conducted in Section 3.3 are available. Consequently, the same experiments were also performed with the sensory data as inputs. Above we revealed already some results. For instance, that we can obtain good prediction performance for gait percent prediction with sensor data as input and even generalization ability. In the following, a more detailed comparison of the prediction results with sensor and motion capturing data is given.

Window-in-Time Setup

First, the results for learning a unique predictor for each gait-speed combination are compared. While doing so, we exemplary use a gait percent predictor for walking at 1.6 m/s. Moreover, it can be differed between the two input-output setups presented in Section 2.3. Starting with the window-in-time approach, where the tunable parameters are the window size, the used input data and the output data. As in Section 3.3, the output data is, for the untransformed case, set to $y \in [0, 100]$. In this case, predictors using only single signals as inputs are, at least for $\tau = 1$, not worth using. More in detail, the error and boundary

peak dimensions are nearly identic to the results for the motion capturing data. To improve prediction results, the predictor is provided with more information. One approach is to keep the window size fixed to $\tau = 1$ and to combine the signals of different sensors to yield new input data. The prediction results for combinations of shank angle and velocity or accelerometer data are equally good for sensor and motion capturing data. In both cases, the usage of more than two of this values leads to no noticeable improvements. When using only accelerometer data or accelerometer data in combination with force data, it is possible to note some changes. Here, the results are not that accurate for the sensor data as for the motion capturing data. The boundary peaks for motion capturing and sensor data, in contrast, stay similar for all input combinations. All in all, the predictor for shank angle and velocity is also for sensor data the most favorable one. Its predictions are accurate and it requires only one sensor.

Another way to provide the predictor with more information, is to increase the window size so that also previous sensor measurements are considered. For $\tau = 1$, for example, the predictor considers only the actual measurement and for $\tau = 5$ the actual measurements plus the last four measurements are used. As for the motion capturing data, the single input predictors and the predictors with accelerometer and force data benefit the most. They become more usable for $\tau = 5$ or at least for $\tau = 10$. Moreover, the error dimensions and boundary peaks are nearly identic for motion capturing and sensor data. Only when using accelerometer data alone (Ax-Ay), an increasing window size results in different error dimensions. E.g., when increasing τ from 1 to 5 and considering the motion capturing data, the prediction error decreases from $\pm 20\%$ to $\pm 4\%$. For the sensor data, in contrast, it decreases from $\pm 50\%$ to $\pm 12\%$. Combinations of gyroscope with accelerometer data show still no relevant improvements when increasing the window size. So, greater τ provide no additional information here. Note that the prediction results for the different input combinations and different τ are given in Table A.1. In general, window size increases introduces delay. If for instance a sudden jump occurs, it takes $\tau - 1$ steps until all the old data is flushed. The computation time increases for larger τ similar as for the motion capturing predictors. Experimenting with the window size reveals that the predictor for shank angle and velocity with $\tau = 1$ is still the favorable one. It gives good prediction performance and depends only on one sensor. In addition window size 1 is used, so that the computation time is as short as possible and sudden jumps can be recognized immediately.

Recurrent Setup

The second approach is the recurrent input-output setup. For performing experiments the window size is fixed to 1 and the last prediction, which is used as additional input, is expected to be in an error range of $\pm 5\%$. If the last prediction exceeds this bound, the predictor might be fooled. All prediction results are shown in Table A.2.Comparing the prediction results to those of the window-in-time setup with $\tau = 1$, all inaccurate predictors improved to usability. These are especially the predictors which use only a single input or accelerometer data only. As for motion capturing data, the improvements result from the additional input which is the last prediction result. Here, the influence of the last prediction is relatively high, so the linear increasing nature of gait percent is explicitly incorporated into the predictors. For combinations of sensor data, in contrast, the additional input does not boost the performance. Consequently, those input combination are already distinguishing enough. In comparison to motion capturing data, the overall error is similar, but some more predictors exceeds the error

bound (±5%) by 1%. Even if the error bound is exceeded only by a small degree, this can cause difficulties like a higher prediction error. During testing, however, no difficulties were observed. For testing purposes, the experiments were also performed with an error bound of ±3%. In this case, the error bound is exceeded by a larger amount of predictors and some higher prediction errors, because of too inaccurate last predictions, were observed. Note that also the recurrent approach is also for the sensor data still prone to the same degree of boundary issues.

Transformed Output

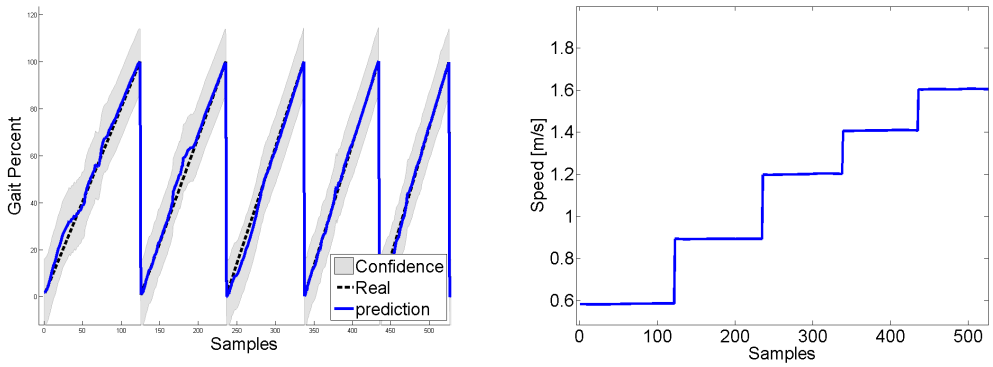
As for motion capturing data, the boundary issues can be eliminated when using a transformation of the output signal. The output signal can either be transformed with a sine or a cosine, where the domain for both can be $[-\pi, \pi]$ or $[-\pi/2, \pi/2]$. For more details see Section 3.3. All transformations give a continuous output signal that has no jumps, expect the sine transformation with $y \in [-\pi/2, \pi/2]$. This transformation, however, reduces the jump in the output signal from 100% to 2%. As for motion capturing data, this only reduces the boundary issues to some degree but does not eliminate them. In contrast to the motion capturing data, the cosine for $y \in [-\pi, \pi]$ is also not able to eliminate the boundary issues. The boundary issues are reduced further (approximately ±30%), but they still persist. So, the boundary peaks are only eliminated when using the sine with $[-\pi, \pi]$ or the cosine with $[-\pi/2, \pi/2]$. Regarding the sine with $[-\pi, \pi]$, the prediction results are, expect the boundaries, not so accurate as for untransformed case. Only for the cosine transformation with $\pi/2$ the prediction results are as accurate as for the untransformed case and consequently this transformation is favorable. For the full results see Table A.3. Note that also for the transformed case the predictor for shank angle and velocity with $\tau = 1$ is preferred. The result of the transformed predictor are transfered to gait percent according to Section 3.3.

Covariance Function

Besides the input-output setup, the covariance function is another subject of change. An possible alternative to the squared exponential covariance function is the Matérn covariance function. Here, the Matérn classes of $\nu = 1/2$, $\nu = 3/2$ or $\nu = 5/2$ are considered. The experiments are performed for untransformed predictors with window size 1. For increasing ν the smoothness of the underlying functions increases. Regarding the prediction results itself, only small details change when increasing ν. The overall prediction performance remains similar. Moreover, the Matérn classes performs equally good as the squared exponential covariance function. Compared to the motion capturing data, the sensor data experiments give the same results.

Speed Independent Gait Percent Prediction

Summarized, the gait percent prediction results for unique speed-gait predictors are very similar for motion capturing and sensor data. The results are accurate and can be used for gait percent prediction with an active ankle prosthesis. Nevertheless, it is more convenient to have a predictor for all speeds of a specific gait. One reasons therefore is the reduction of needed predictors. Another reasons is that gait percent prediction will no longer depend on speed prediction. If such a dependency is present, speed prediction errors can cause gait percent prediction errors. As revealed when introducing the transfer function for deriving shank angle, the prediction results generalize when using a predictor for all speeds. As for



(a) Gait percent prediction for five different steps

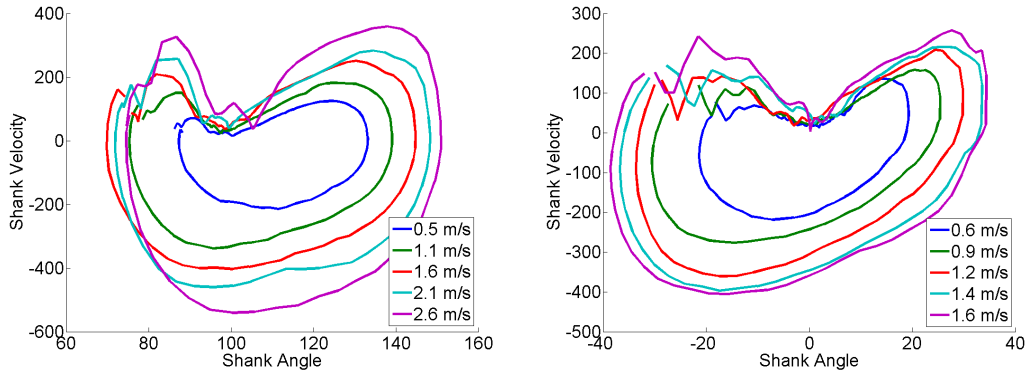
(b) Speed each step is performed at

Figure 4.2.: (a) gait percent prediction performance for five different steps. The used predictor is based on Gaussian process regression, shank angle and velocity as input and a window-in-time setup with $\tau = 1$. Each steps corresponds to a different speed. The precise speeds are given in Panel (b). Note that each speed produced by the treadmill fluctuates a little

motion capturing data, the prediction error increases a little when generalizing but it is still accurate enough for real world applicability. In case of the transformed predictor with shank angle and velocity as input and $\tau = 1$ the overall prediction error is $\pm 4\%$ for a specific gait-speed combination. For all speeds the prediction error increases to $\pm 5\%$. The prediction results are shown in Figure 4.2. This figure displays 5 steps, where each step corresponds with one of the considered speeds of walking. Starting with the slowest speed and increasing to the fastest one. Note that the computation time for a predictor that uses all speeds can be reduced by using fewer samples for each speed. All in all, the prediction time for ones gait cycle can be equal to the time needed by a specific gait-speed predictor with the same input signals and window size.

4.3 Speed Prediction

This section starts with performing the speed prediction experiments of Section 3.4 with sensor instead of motion capturing data. The results of both input sources are compared to evaluate if the prediction methods generalize to sensor data and, therefore, are transferable to the active ankle prosthesis. Regarding the experiment's setup of both data sources, a major difference becomes obvious. For the sensor data, the different speeds are closer together and are recorded one after each other. So, speed transitions can be inspected as well. After comparing the results, another approach for speed prediction is introduced. This approach is feature based and is superior to the already given ones. In context of supervised machine learning, a feature is computationally derived from the original input data provided to the predictor. Therefore, a function $\phi(x)$ is applied to the given data x. If the function value gives, in context of the considered problem, more information than the original value, the prediction performance improves.



(a) Motion capturing data **(b)** Sensor data

Figure 4.3.: Panel (a) depicts the shank angle-velocity cyclogram based on the motion capturing data. In panel (b) the cyclogramm is computed from the shank angular velocity provided by the inertial measurement unit. Therefore, a transfer function is applied to give shank angle. In comparison to Panel (a), the shank ankle computed from the sensor data shows different scales and is centered around zero.

4.3.1 Comparison to Motion Capturing

Before the speed prediction experiments are performed for the sensor data as input, it must be noted that we cannot directly compare the results to those obtained for motion capturing data. This has several reasons. As hinted above, one reason is given by the fact that the database for the sensory data is not identical with the motion capturing database. In case of sensor data, the recorded speeds are more fine grained. The speeds are separate by a distance of just $0.2m/s$ and not by $0.4m/s$ as for motion capturing data. Consequently, it might be more difficult to discriminate between speeds, which in turn might decrease prediction performance. Another reason is constituted by the input signals themselves. One the one hand the signals are noisier, on the other hand the shank angle is derived with the use of a transfer function. The transfer function causes a similar shape as for the real shank angle, but gives different absolute values. The differences are noticeable when comparing both angle-velocity cyclograms given in Figure 4.3.

Even though this differences did not influence gait percent prediction, their influence on speed prediction needs to be considered and analyzed as well.

Speed Classification

The first experimental setup is the unlimited support vector machine for a window-in-time approach. Unlimited means that the same support vector machine is used for the whole gait cycle and therefore speed prediction is independent of gait percent prediction. The speed predictor, however, depends on gait prediction, since an unique support vector machine is used for each gait. When comparing the prediction results to those for motion capturing data, several similarities become apparent. First, for all input signals and their combinations the accuracy increases with increasing τ. Second, the relation between the performance of different signal combinations is the same for sensor and motion capturing data. The single input classifiers are still not worth using, sensor combinations containing gyroscope data (shank angle and velocity) show higher performance than others combinations and force data still

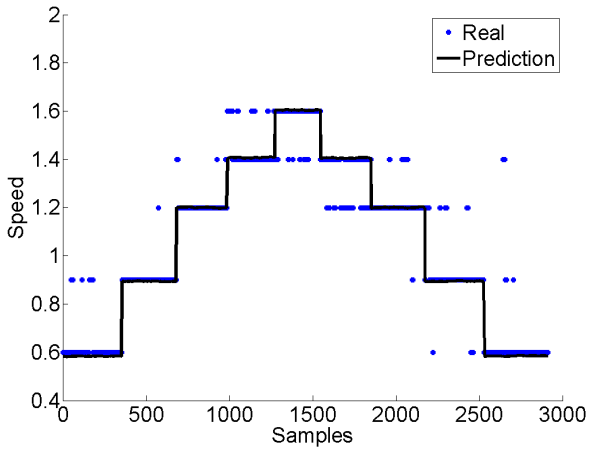
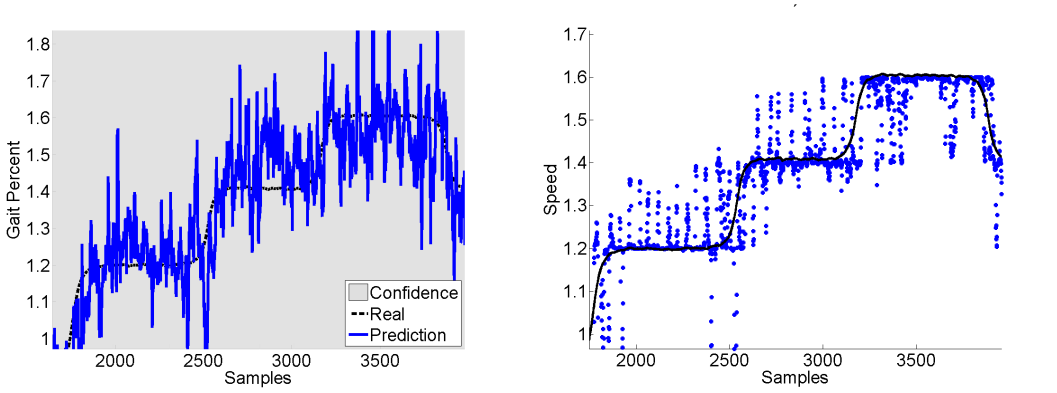


Figure 4.4.: Speed prediction results for a support vector machine with $\tau = 1$ and shank, accelerometer and force data as input.

boosts performance. Third, it is also required to use $\tau = 10$, since predictors with a smaller window size are not applicable for active ankle control. For τ equals 10, the predictor which uses all input signals as input still performs the best. Its prediction results are displayed in Figure 4.4. This Figure reveals that the predictor produces often high numbers of consecutively wrong classified speeds. Even with the use of a voting scheme this cannot be corrected for. Furthermore, the overall numbers of misclassification is that high, that the predictor is also not applicable for active ankle control. This becomes also obvious when comparing the classification accuracies. The considered predictor gives only an accuracy of 88% which is not sufficient. The same predictor for the motion capturing data, in contrast, achieves with an accuracy of 96 good results. Comparing also the accuracies of other configurations, which are given in Table A.4, reveals that the predictors for the sensor data perform generally worse than those for motion capturing data. More in detail, the accuracies for the sensor data, are about $5 - 12\%$ below those of the motion capturing data. As hinted earlier, that might be because the speeds are more close and probably harder to distinguish or due to the transfer function applied for obtaining the shank angle. Even though the angle-velocity cyclogram for motion and sensor data (compare Figure 4.3) are different in absolute shank angle, the influence of the transfer function seems negligible. This becomes evident, by comparing the prediction results of classifiers that do not require shank angle as input. For instance, the predictor for shank velocity as input and $\tau = 10$, achieves for the motion capturing data an accuracy of 71% and for the sensor data an accuracy of 60%. So, the unlimited classifier shows some generalization to sensor data but is for these fine grained speeds, even for large τ, not applicable for active ankle control.

Regression

As seen for motion capturing data, classification is for the task of speed prediction preferable to Gaussian process regression. This is obvious, since speed prediction is in the considered setup rather a classification than a regression problem. However, Gaussian process regression is evaluated for the sensor data as well. The evaluation is not only performed for the sake of completeness, but also for comparing support vectors machines and Gaussian process regression in case of speed transitions. The motion capturing database did not comprise transitions between speeds, the sensor database, however, does. Before the transition behav-



(a) Transition performance when using Gaussian process regression

(b) Transition performance when using classification

Figure 4.5.: Prediction performance for speed transtions. Both predictors use a window in time approach with $\tau = 10$ and shank, accelerometer and force data as input. In panel (a) Gaussian process regression is used, and in (b) support vector machines are applied.

ior is analyzed, the performance of sensor and motion capturing data is compared. From an error point of view, the predictors for both input signals are very similar. The best performance is achieved for the larger window sizes and is given by predictors that show errors bounded to $\pm 0.4 m/s$ or $\pm 0.3 m/s$. All prediction errors for the Gaussian process setup are given in Table A.5. Note that additional increases of τ cannot boost the performance any further. The relation between the performance of different signal combinations is the same for sensor and motion capturing data. The reported errors of $\pm 0.4 m/s$ or $\pm 0.3 m/s$ are also realistic for the motion capturing data. For this kind of data, however, the distance between the considered speeds is larger. An error of $\pm 0.4 m/s$ is in case of motion capturing data only the misclassification by one speed. For the sensor data, the prediction is wrong by two speeds. Furthermore, the prediction results or rather the underlying function varies always up and down, so that errors happen relatively often. All in all, the Gaussian process predictors are for sensor data even worse than for motion capturing data.

The data for speed transitions is not included into the training data. Including data for all possible transitions is nearly infeasible, since speed transitions can happen at every gait percent. Even to record some of those transitions would require a lot of precision and endurance. To evaluate how the Gaussian process predictors perform for transitions, the best predictor, namely the one for angle-vel-Ax-Ay-fzl as input and $\tau = 10$, is used. Its prediction performance is shown in Figure 4.5a. As for constant speeds, its performance is also poor for transitions. The performance for support vector machine predictors is evaluated with the same predictor configuration. To predict intermediate speeds, the interpolation technique presented in Section 2.2.2 is used. Also for classification the results are not that good, as can be seen in Figure 4.5b.

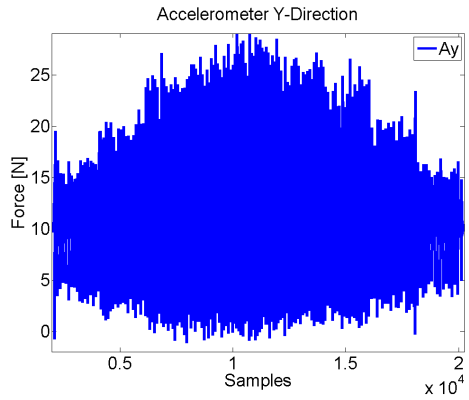
Limited Predictors

Both evaluated setups are not accurate enough for real world applicability. Section 3.4 showed that the prediction performance can be improved by dividing the gait cycle into gait percent areas and using an unique speed predictor for each area. Consequently, we try to improve speed discrimination for sensor data also with these technique. In this case it might

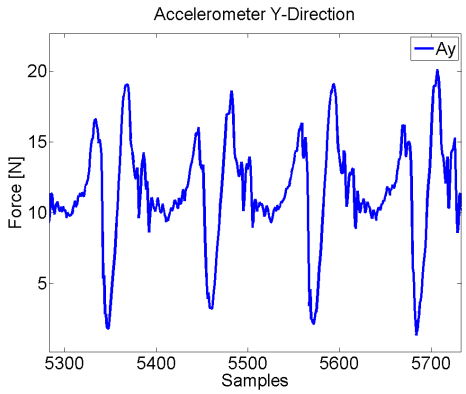
be more difficult, since the different speeds are closer. As size for the gait percent areas 10 gait percent is a realistic span, since the gait percent prediction error is still within ±5%. The prediction performance of all different gait percent areas is accessed with the use of the two representative areas given in Section 3.4. Even though the shank angle changed a little due to the transfer function, both areas are still representative. The first area lasts from 10 to 20 gait percent and represents those sections where the input signals are close. The area 60–70 gait percent represents the ones where the input data is easier to distinguish. The results for both areas are shown in Table A.6. As for motion capturing data, limiting the predictors to specific gait percent areas, improves the overall prediction performance compared to the unlimited case. The relationship between the performance of different input signal combinations stays also the same. However, there are also differences with respect to the motion capturing data. For the predictors concerned with 10–20 gait percent and $\tau = 1$, the predictions are worse than for the unlimited case. Consequently, the considered speeds are to close to be distinguished with a window size of 1. Moreover, we need to keep in mind that the accuracy for the unlimited case is constituted by the prediction results for the whole gait cycle. Besides such complicated areas like 10–20 gait percent, also more distinguishable areas are covered and may boost the accuracy. For larger window sizes, more information are given to the classifier, so that the accuracy is approximately $5-15\%$ higher as for the unlimited case. The information given by larger window sizes are nevertheless not sufficient enough for achieving accuracies above 90%. Only the classifier for angle-vel-ax-ay-fzl and $\tau = 10$ gives exactly 90% accuracy. In case of 60–70 gait percent, the predictors achieve already for $\tau = 1$ accuracies superior to their unlimited pendants. Also for larger window sizes the accuracies stay superior, in detail, the accuracies are about 10–20% above the unlimited case. The best predictor is constituted by angle-vel-Ax-Ay-fzl as input and gives for $\tau = 5$ and $\tau = 10$ accuracies of 89% and 93%. Compared to the area of 10–20 gait percent, predictors for 60–70 gait percent perform better because the input data is easier to distinguish. The accuracies are also for 60–70 gait percent smaller than for the identical motion capturing predictors, since the speeds are a bit closer. Speed transitions are not considered for predictors restricted to special gait percent areas, since we do not have the transitions for all areas to test for. As stated earlier, the construction of such a database is not feasible.

4.3.2 Feature-Based Classification

For motion capturing, the usage of restricted predictors introduced a dependency to gait percent prediction but improved speed prediction to applicability. In case of active ankle control, real world applicability is given when accuracies above 90 close to 100% are achieved. Such good accuracies were only partially achieved for the sensor data, because the speeds are closer together. So, a better performing methods needs to found. One way to improve performance is to introduce a feature which gives additional information to the predictor. Therefore, a function $\phi(x)$ is introduced that computes the feature given the data originally provided to the predictor (x). Note that it is not straight forward to define features that boost prediction performance. We introduce a feature inspired by inertial measurement unit based methods for walking speed and stride length estimation [24]. Such methods apply integration, in addition with tweaks like estimating the accelerometer orientation and transferring it into a global coordinate system. Here, a transfer function is applied to give a feature that gives, like stride length, valuable information for determining the intended speed. As in-



(a) Accelerometer data (y-direction) for the scenario depicted in Figure 4.1b.



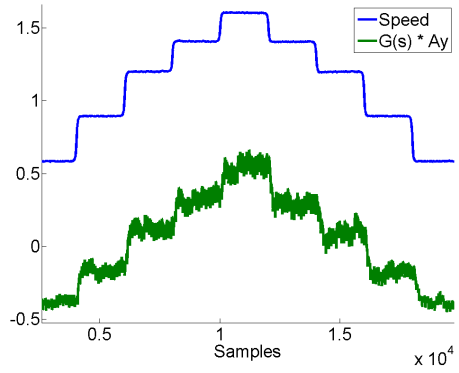
(b) Zoomed in view on the accelerometer data.

Figure 4.6.: (a) accelerometer data in y-direction while increasing speed from $0.6m/s$ to $1.6m/s$. In between a fixed number of speeds is considered, whereat each speed is walked for 20 seconds. After reaching $1.6m/s$, the speed is decreased again. (b) shows a zoomed-in view of the accelerometer data. In more detail, 4 consecutive steps are shown.

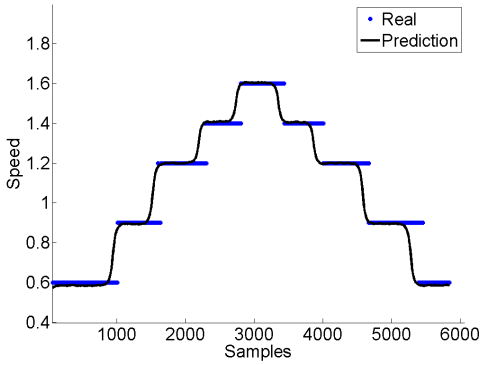
put to the transfer function the accelerometer measurements in y-direction are used. The accelerometer measurements for the conducted walking experiments (compare Figure 4.1b) are displayed in Figure 4.6a. Regarding the walking experiments, the speeds 0.6, 0.8, 1.0, 1.2, $1.6m/s$ were walked in consecutive order. Each speed was given by the treadmill for 20 seconds. After reaching $1.6m/s$, the same procedure was repeated in decreasing order. The displayed accelerometer measurements in y-direction reveal some trend according to these experiments. This trend is visible in signal height which increases with walking speeds. If the accelerometer signal is viewed in detail, this trend is no longer obvious (see Figure 4.6b). Instead the cyclic nature of gait becomes evident. To extract the trend and to suppress the high frequencies, a transfer function was designed. This transfer is given by

$$\frac{\alpha}{(\tau s + 1)^2}, \tag{4.2}$$

where α and τ are non-negative, free parameters. The transfer function is stable and τ influences its fastness. α is an amplification factor. Both parameters imply how speed changes are visible and how big the differences between speeds are. It must be found a compromise between both. If the transfer function is too fast, the differences between speeds are not visible anymore and discrimination cannot be achieved. Applying an optimized transfer function to the accelerometer data in y-direction, gives the results shown in Figure 4.7a. The computed feature is similar to speed and most likely will enable speed classification. Furthermore, it was found that applying the transfer function also to the accelerometer signal in x-direction, gives additional information for speed prediction. So, a speed predictor for the input combination of $G(s)Ay$ and $G(s)Ax$ is learned and used. The predictor is unlimited and uses a window size of 1. The learned predictor is tested by applying the transfer function to the incoming accelerometer data and using the resuling values as input to the predictor. The prediction results are shown in Figure 4.7b. Note that the results are nearly perfect and that this predictor outperforms all previously presented approaches. Furthermore, it was tested if the transfer function is fast enough in reflecting sudden speed changes. A test scenario was



(a) Comparison of walking speed and transfer function output

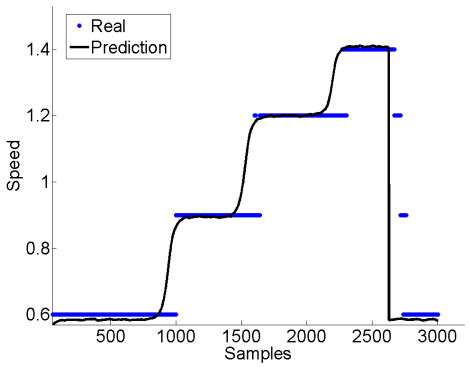


(b) Classification results when using the transfer function results as input

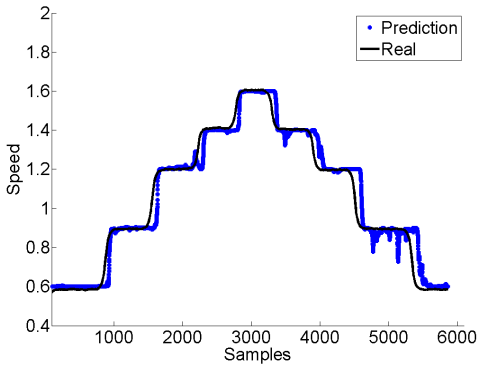
Figure 4.7.: Panel (a) compares the walked speed with the results obtained by applying the presented transfer function to the corresponding accelerometer signal in y-direction. Speed and transfer function result are similar in shape and, consequently, the transfer function enables classification results as shown in panel (b). As classifier a support vector machine with $\tau = 1$ is used. The input is given by combining the results of transfer function applied to accelerometer signal in y and in x direction.

constructed where the speed increases until $1.6m/s$, afterwards it drops suddenly (without transition) to $0.6m/s$. The prediction results are given in Figure 4.8a. As can be seen, the transfer function can deal with such sudden changes. It even detects some speeds in between before detecting the speed we jumped to. This is more convenient for the prosthesis itself, because it is not possible to change between speeds that fast as given by the training data. Moreover, this period with speeds is with an length of about one step relatively short.

Concluding the transitions between speeds are examined again. Figure 4.7b showed good results, but transitions were ignored. As long as a transition takes place the old speed is detected. Once the next speed is reached it is correctly classified. Due to the good results, it



(a) Classification results (setup as for Figure 4.7b) for an abrupt change in speed.



(b) Interpolation between speeds.

Figure 4.8.: Panel (a) shows the speed prediction performance for a support vector machine with $\tau = 1$ and $G(s)Ay$ and $G(s)Ax$ as input. As can be seen, the transfer function and, therefore, also the predictor is able to adapt to sudden jumps in speed. In Panel (b) the same predictor is used. Instead of giving the pure classification results as in Figure 4.7b, the interpolation between speeds is shown.

might also be possible to interpolate between speeds as described in 2.2.2. It turns out, that interpolation is possible. The results are depicted in Figure 4.8.

4.4 Supervisory Controller

In Section 3.6 two supervisory controllers were proposed based on evaluations performed with motion capturing data. In more detail, the motion capturing data was used to compute sensor-like data for gait, speed and gait percent prediction. For each task different setups were executed and compared. The best predictors were used to construct two distinct supervisory controllers. Each controller demonstrates that supervised machine learning techniques are feasible for active ankle control. The operating prosthesis, however, gives sensor and not motion capturing data. Compared to sensor data, motion capturing data is more accurate and noise free. This chapter demonstrated that the introduced machine learning concepts also apply on the prosthesis. In the following, the results for sensor data are summarized shortly. Afterwards, we purpose a controller applicable for active ankle control based on sensor data.

Gait percent prediction scaled perfectly to sensor data. The prediction results are as those given in Section 3.6. Consequently, the transformed predictor with shank angle and velocity as input and windows size 1 is the most promising one. Note that the gyroscope gives only shank velocity but not shank angle. To derive shank angle a transfer function introduced in [11] is used. Even though speed prediction scales to sensor data, all predictors achieve lower accuracies. This is due to the different experimental setups. In case of sensor data, only level-ground walking is examined and the considered speeds are closer together. The distance between two different speeds decreased from $0.4m/s$ for motion capturing data to $0.2m/s$. So, we can observe that closer speeds are harder to distinguish. The corresponding decrease in accuracy rendered the evaluated speed predictor less usable for active ankle control. That is why, we designed two features that improves speed prediction. Both features are computed by applying a transfer function to either the accelerometer signal in x or in y direction. The classifier based on this features performs really accurate for a window size of one.

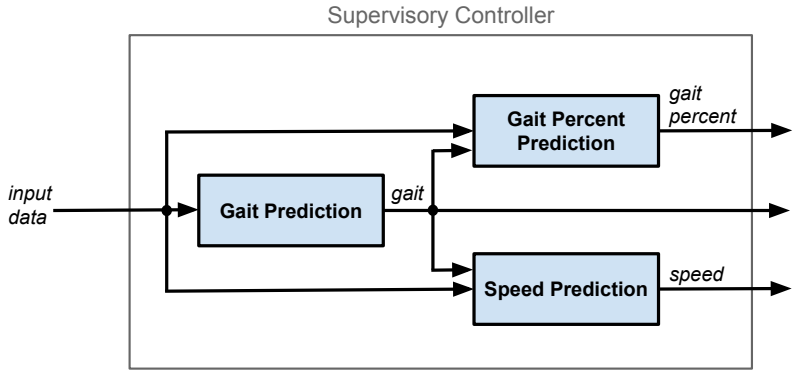


Figure 4.9.: Structure of the proposed supervisory controller. At fist gait is predicted. Afterwards gait percent and speed prediction can take place. Note that the controller's structure is the same as for the one based on motion capturing data given in Figure 3.15a. However, the version based on sensor data is a way more efficient, since all predictors use small window sizes and are accurate.

Summarizing Chapter 4, supervised machine learning methods are applicable for designing a supervisory controller for an active ankle prosthesis. The controller uses the prosthesis's sensor measurements as inputs and gives accurate predictions for gait, speed and gait percent. Since speed prediction was improved with the introduced features, the structure of the supervisory controller is as in Figure 4.9.

First the gait mode is predicted. When gait is known, speed and gait percent can be predicted independently. So, the controller has two benefits when compared to the two controllers presented in Section 3.6. First, the predictors only depend on gait prediction and nothing more. Second, both predictors perform already with $\tau = 1$ and require just one type of sensor measurement. In case of speed prediction its accelerometer data and for gait percent prediction gyro data.

5 Outlook

The previous sections evaluated different machine learning setups in context of active ankle control. As result a supervisory controller for a powered ankle prosthesis was proposed. At first, this section summarizes how the introduced concepts contributed to the design of the proposed controller. Afterwards, future work is presented.

5.1 Conclusion

This work introduced recent supervised machine learning techniques for control of active prosthetic devices. The device considered here is an active ankle prosthesis equipped with an inertial measurement unit mounted at the shank. Note that the introduced techniques can most likely be applied for other devices and sensor configurations as well. The sensor values produced by the inertial measurement unit are used to infer the user's intent. Based on the intent, we adapt the prosthesis's desired trajectory to achieve best possible user support. In this case, the intent is given by gait, speed and gait percent. All values are predicted by an introduced supervisory controller which relies on supervised machine learning methods such as Gaussian process regression and support vectors machines. If gait, speed and gait percent predictions are available, the desired nut position can be determined by a lookup. To enforce the desired trajectory a slave controller, here a PD controller, is used.

The supervisory controller was designed and implemented based on data recorded by a motion capturing system. Since such data is accurate and noise free, it is perfect for evaluating the feasibility of supervised machine learning methods in context of the considered prediction tasks. The motion capturing data was recorded by a healthy male subject of 1.86 m height and 76 kg weight. For the recordings a treadmill was used and trials for different constant walking and running speeds were performed. After recording the data, it was post-processed to yield data as would be obtained from the inertial measurement unit. In addition, force data from the treadmill can be used as it were recordings from a force sensor. For each prediction tasks different setups like window-in-time or recurrent approaches were evaluated. Moreover, all tasks were fine-tuned, e.g. the usage of a transformed output for gait percent prediction, to achieve optimal performance. The best predictors for the tasks of gait, speed and gait percent prediction were chosen to compose a supervisory controller. In case of speed prediction, two predictors are possible. The first predictor gives simplicity and independence, whereas the second predictor is more complex but in some cases faster and more accurate. Consequently, not one but two supervisory controllers were proposed, differing just by means of speed prediction. Both supervisory controllers demonstrate that it is feasible to design an accurate controller based on supervised machine learning methods.

Because the controllers are based on the conducted motion capturing experiments, they can only indicate performance for the real system. To demonstrate that the introduced concepts also apply on the prosthesis, we performed evaluations with real sensor data as well. Therefore, a sensor database was generated by the same subject on the same treadmill. The database contains only level-ground walking trials for different speeds. Note that

the considered speeds are closer together than for the motion capturing data. Gait percent prediction gives for sensor data performance as good as for the motion capturing data. Speed prediction, however, is not identic for motion capturing and sensor data. Comparing different speed predictors, the performance differences stays the same, but overall the predictors give reduced performance. This is exactly what we expected to observe when reducing the distances between the considered speeds. The closer the different speeds, the harder to distinguish between them. By showing that predictors for motion capturing and sensor data give similar results, we demonstrated that motion capturing data lends itself perfectly for feasibility studies in context of machine learning methods. Instead of buying different sensors and performing several recordings, we can evaluate the impact of specific data by computing it from motion capturing recordings.

As speed prediction was for the motion capturing experiments only slightly above the level of applicability, it decreased below this level when performing the same experiments for the closer speeds given by the sensor data. That is why we introduced two features that boosts speed prediction performance dramatically. One feature is computed by applying a transfer function to the accelerometer signal in y direction. The other feature is given by using the same transfer function with accelerometer data in x direction as input. With those features we can classify speeds with an accuracy close to 100%.

As overall result we proposed the supervisory controller for active ankle control given in Figure 4.9. The controller is due to small window sizes fast and moreover accurate. For gait percent prediction the error is bounded to $\pm 5\%$. In case of speed and gait prediction accuracies close to 100% and 95% are achieved, respectively. Moreover, the small window sizes allow the controller to perform even when initialized at a random point in gait or when abrupt changes in motion occur.

5.2 Future Work

In future work the proposed supervisory controller can be extended and evaluated in several ways. For evaluation, a clinical study with amputees can be conducted. At first, it could be evaluate how the controllers performs or rather feels compared to existing control approaches. Another evaluation objective is how the control approach generalize to different subjects or rather users. If the controller doe not scale at all, it can be extended with an mechanism for user adaption, e.g. by learning several predictors and choosing the one the user is closest to. Moreover, we want to cover all situations an active prosthesis in naturally used in. Therefore the controller needs to be extended to more different gaits such as stairs ascent and descent or incline and decline walking.

As demonstrated above, motion capturing data is perfect to study feasibility in context of active prosthetic device control based on machine learning approaches. For instance it can be used to evaluate which mechanisms perform good for a given prediction tasks and what input signals give valuable information. So, another future task is the implementation of a motion capturing based evaluation framework. The framework should allow to compute different sensors values given motion capturing data and provided different machine learning methods. All in all, the framework can facilitate and easy evaluation of even more different approaches and help to gain some valuable insights about signals relevant for human motion.

Bibliography

[1] G. Heller, C. Günster, and E. Swart. Über die häufigkeit von amputationen unterer extremitäten in deutschland. In *Deutsche Medizinische Wochenschrift*, 130(28/29), pages 1689–1690. Thieme, 2005.

[2] C. Mayer and W. Siems. *100 Krankheitsbilder in der Physiotherapie*. Springer, 2011.

[3] G. Heller, C. Günster, and H. Schellschmidt. Wie häufig sind diabetes-bedingte amputationen unterer extremitäten in deutschland? In *Deutsche Medizinische Wochenschrift*, 129(9), pages 429–443. Thieme, 2004.

[4] J. E. Shaw, R. A. Sicree, and P. Z. Zimmet. Global estimates of the prevalence of diabetes for 2010 and 2030. *Diabetes Research and Clinical Practice*, 87, 2010.

[5] S.K. Au, H. Herr, J. Weber, and E.C. Martinez-Villalpando. Powered ankle-foot prosthesis for the improvement of amputee ambulation. In *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference on*, pages 3020–3026, Aug 2007.

[6] Grabowski and D'Andrea. Effects of a powered ankle-foot prosthesis on kinetic loading of the unaffected leg during level-ground walking. *Journal of NeuroEngineering and Rehabilitation 2013*, 2013.

[7] S.K. Au, J. Weber, and H. Herr. Biomechanical design of a powered ankle-foot prosthesis. In *Rehabilitation Robotics, 2007. ICORR 2007. IEEE 10th International Conference on*, pages 298–303, June 2007.

[8] Thomas G. Sugar Kevin W. Hollander, Robert Ilg and Donald Herring.

[9] M. Eslamy, M. Grimmer, S. Rinderknecht, and A. Seyfarth. Does it pay to have a damper in a powered ankle prosthesis? a power-energy perspective. In *Rehabilitation Robotics, 2013. ICORR 2013. 13th International Conference on*, Juni 2013.

[10] H.A Varol, Frank Sup, and M. Goldfarb. Multiclass real-time intent recognition of a powered lower limb prosthesis. *Biomedical Engineering, IEEE Transactions on*, 57(3):542–551, March 2010.

[11] M.A Holgate, T.G. Sugar, and AW. Bohler. A novel control algorithm for wearable robotics using phase plane invariants. In *Robotics and Automation, 2009. ICRA 2009. IEEE International Conference on*, pages 3845–3850, May 2009.

[12] Flowers W.C. Grimes D.L. and Donath M.

[13] He Huang, T.A Kuiken, and R.D. Lipschutz. A strategy for identifying locomotion modes using surface electromyography. *Biomedical Engineering, IEEE Transactions on*, 56(1):65–73, Jan 2009.

[14] S.K. Au, P. Bonato, and H. Herr. An emg-position controlled system for an active ankle-foot prosthesis: an initial experimental study. In *Rehabilitation Robotics, 2005. ICORR 2005. 9th International Conference on*, pages 375–379, June 2005.

[15] Francesco V. Tenore and R. Jacob Vogelstein. Revolutionizing prosthetics: Devices for neural integration. *JOHNS HOPKINS APL TECHNICAL DIGEST*, 30(3).

[16] H.A Varol and M. Goldfarb. Decomposition-based control for a powered knee and ankle transfemoral prosthesis. In *Rehabilitation Robotics, 2007. ICORR 2007. 10th International Conference on*, pages 783–789, June 2007.

[17] Samuel K. Au, Jeff Weber, and Hugh Herr. Powered ankle-foot prosthesis improves walking metabolic economy. *Journal of Robotics and Automation*, 25(1):51–66, February 2009.

[18] M.F. Eilenberg, H. Geyer, and H. Herr. Control of a powered ankle foot prosthesis based on a neuromuscular model. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 18(2), April 2010.

[19] Eilenberg MF Endo K Barnhart C Herr H Markowitz J, Krishnaswamy P. Speed adaptation in a powered transtibial prosthesis controlled with a neuromuscular model. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 2011.

[20] H.A. Varol and M. Goldfarb. Real-time intent recognition for a powered knee and ankle transfemoral prosthesis. In *Rehabilitation Robotics, 2007. ICORR 2007. 10th International Conference on*, pages 16–23, June 2007.

[21] B.E. Lawson, H.A Varol, and M. Goldfarb. Standing stability enhancement with an intelligent powered transfemoral prosthesis. *Biomedical Engineering, IEEE Transactions on*, 58(9):2617–2624, Sept 2011.

[22] Frank Sup, H.A Varol, and M. Goldfarb. Upslope walking with a powered knee and ankle prosthesis: Initial results with an amputee subject. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 19(1):71–78, Feb 2011.

[23] A.W. Boehler, K. W. Hollander, T.G. Sugar, and Dosun Shin. Design, implementation and test results of a robust control method for a powered ankle foot orthosis (afo). In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 2025–2030, May 2008.

[24] Shuozhi Yang and Qingguo Li. Inertial sensor-based methods in walking speed estimation: A systematic review. *Sensors*, 12(5):6102–6116, 2012.

[25] Yoonseon Song, Seungchul Shin, Seunghwan Kim, Doheon Lee, and K.H. Lee. Speed estimation from a tri-axial accelerometer using neural networks. In *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, pages 3224–3227, Aug 2007.

[26] Shinji Miyazaki. Long-term unrestrained measurement of stride length and walking velocity utilizing a piezoelectric gyroscope. *Biomedical Engineering, IEEE Transactions on*, 44(8):753–759, Aug 1997.

[27] Büla C Leyvraz PF Robert P Aminian K, Najafi B. Spatio-temporal parameters of gait measured by an ambulatory system using miniature gyroscopes. *Journal of Biomechanics*, 35(5):689–699, 2002.

[28] J.C. Alvarez, R.C. Gonzalez, D. Alvarez, AM. Lopez, and J. Rodriguez-Uria. Multisensor approach to walking distance estimation with foot inertial sensing. In *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, pages 5719–5722, Aug 2007.

[29] Sethu Vijayakumar, Aaron D'souza, Tomohiro Shibata, Jörg Conradt, and Stefan Schaal. Statistical learning for humanoid robots. *Autonomous Robots*, 12(1):55–69, 2002.

[30] Hal varian on how the web challenges managers. URL `http://www.mckinsey.com/insights/innovation/hal_varian_on_how_the_web_challenges_managers`, January 2009.

[31] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[32] Duy Nguyen-Tuong and J. Peters. Local gaussian process regression for real-time model-based robot control. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 380–385, Sept 2008.

[33] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[34] John Reid. What are gaussian processes? URL `https://pythonhosted.org/infpy/gps.html`, 2012.

[35] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

[36] JP Vert, K Tsuda, and B Schölkopf. *A Primer on Kernel Methods*, pages 35–70. MIT Press, 2004.

[37] John C. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical report, Advances In Kernel Methods - Support Vector Learning, 1998.

[38] E. P. Costa, A. C. Lorena, Carvalho, and A. A. Freitas. A review of performance evaluation measures for hierarchical classifiers. In *2007 AAAI Workshop, Vancouver*. AAAI Press, 2007.

[39] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

[40] S. M. Ross. Expectation of a random variable. In *Introduction to probability models*, 2010.

[41] Jeffrey Hausdorff. Gait variability: methods, modeling and meaning. *Journal of Neuro-Engineering and Rehabilitation*, 2(1), 2005.

[42] M. L. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer Series in Statistics, 1999,.

[43] Alan Hreljac, Rodney Imamura, Rafael F Escamilla, Jeffrey Casebolt, and Mitell Sison.

A Prediction Results for Sensor Data

In Section 3 we designed a supervisory controller for active ankle control based on motion capturing data. In doing so, different setups for gait percent, speed and gait prediction were evaluated. The prediction performance of each of those setups is summarized in an own table. Section 4 shows that the predictors used for active ankle control also scale for real sensor data. Moreover, some differences between the prediction results for motion capturing and sensor data are described. To facilitate a detailed comparison, the same tables as in Section 3 are given here for sensor, instead of motion capturing, data.

A.1 Gait Percent

Table A.1.: Performance of gait percent prediction while walking at $1.6m/s$. All predictors are based on Gaussian process regression with a window-in-time setup. The tables shows the prediction errors for varying window sizes. The first row for each window size is the error(+/-) in gait percent, whereas the second row is the boundary error (+/-) in gait percent. The column headings state which input signal combination is used for the corrsponding predictor. Shank angle and shank velocity are abbreviated with *angle* and *vel*. For the acceleration signal in x and y direction *Ax* and *Ay* are used, respectively. The force data of the left foot is denoted with *fzl*.

τ	angle	vel	Ax	Ay	fzl
1	40/40	40/40	45/45	40/45	40/40
	60/60	80/50	60/80	60/60	60/40
5	4/4	8/8	15/10	20/20	20/22
	50/50	85/85	60/60	40/40	30/30
10	3/4	6/6	7/7	10/12	15/15
	40/40	70/80	30/30	40/40	60/50

τ	angle-vel	Ax-Ay	angle-vel-fzl	Ax-Ay-fzl	Ax-Ay-fzl Ax-Ay	angle-vel-Ax-Ay-fzl
1	4/4	50/50	4/4	15/15	4/3	4/3
	40/40	60/80	40/50	70/65	85/90	85/90
5	4/3	12/12	4/4	5/5	4/3	4/3
	40/40	60/80	60/40	80/90	40/80	80/25
10	3/3	5/4	2/3	4/4	3/3	3/3
	60/60	10/11	70/80	40/60	60/60	75/30

Table A.2.: Prediction results for the recurrent setup with $\tau = 1$. For walking at $1.6m/s$, the prediction errors in gait percent and the predictor's lengthscales are given. All lengthscales are ordered by the input signal order given in each column heading. For the input signal's abbreviations see Table A.1. The last lengthscale which is gray correpsonds to the input from the last gait percent prediction.

	angle	**vel**	**Ax**	**Ay**	**fzl**
Error (+/−)	3/3	6/4	6/6	6/6	6/6
Boundary Error (+/−)	40/80	60/50	90/80	50/80	60/60
Lengthscales	0.25	0.37	0.21	19.3	0.63
	0.30	1.10	0.18	0.49	1.30

	angle-vel	**Ax-Ay**	**angle-vel-fzl**	**Ax-Ay-fzl**	**angle-vel-Ax-Ay**	**angle-vel-Ax-Ay-fzl**
Error (+/−)	1/3	5/5	3/2	5/4	2/2	2/2
Boundary Error (+/−)	50/40	90/70	50/50	60/60	80/90	85/85
Lengthscales	0.38,	0.14,	0.41,	0.20,	3.31,	3.98,
	0.43,	4.20,	0.44,	1.64,	0.81,	0.96,
	2.00	0.34	1.32,	1.16,	0.81,	0.96,
			6.98	0.67	135	57.5
					1.42	3.67,
						0.84

Table A.3.: Comparison of different output transformations in context of gait percent prediction while walking at $1.6m/s$. For each transformation($t(y)$), output range and input combination the overall error(+/-) in gait percent is given. As input setup we used the window-in-time approach with $\tau = 1$. The input combinations are abbreviated as in Table A.1.

t(y) - range	**angle**	**vel**	**Ax**	**Ay**	**fzl**
cos - $\pi/2$	20/30	50/55	60/60	60/60	40/60
sin - $\pi/2$	90/100	100/80	100/100	100/100	150/50
cos - π	30/80	95/95	150/150	60/60	90/30
sin - π	50/70	80/50	100/120	120/140	60/60

t(y) - range	**angle-vel**	**Ax-Ay**	**angle-vel-fzl**	**Ax-Ay-fzl**	**angle-vel-Ax-Ay**	**angle-vel-Ax-Ay-fzl**
cos - $\pi/2$	5/4	50/60	5/3	20/35	4/4	5/3
sin - $\pi/2$	80/70	90/95	90/90	80/90	85/90	90/90
cos - π	55/40	95/95	20/25	30/20	20/15	15/150
sin - π	7/6	70/75	6/5	20/30	5/6	5/5

Table A.4.: Varying widow size for unlimited speed prediction based on support vector machines and a window-in-time input-output setup. Each cell gives the accuracy in percent. The corresponding heading denotes the used input signals. For more information on the input signals see Table A.1.

τ	angle	vel	Ax	Ay	fzl
1	30	36	30	33	31
5	60	47	37	39	43
10	61	60	47	51	57

τ	angle-vel	Ax-Ay	angle-vel-fzl	Ax-Ay-fzl	angle-vel-Ax-Ay	angle-vel-Ax-Ay-fzl
1	63	34	68	44	66	71
5	68	55	84	74	73	84
10	71	66	88	84	80	88

Table A.5.: Varying widow size for speed prediction based on Gaussian process regression and a window-in-time input-output setup. Each cell gives the prediction error(+/-) in m/s. Note that the input signals are abbreviated as in Table A.1

τ	angle	vel	Ax	Ay	fzl
1	0.6/0.4	0.8/0.6	0.8/0.6	0.8/0.6	0.7/0.5
5	0.6/0.4	0.6/0.5	0.8/0.6	0.6/0.4	0.6/0.6
10	0.5/0.5	0.5/0.5	0.7/0.5	0.4/0.5	0.6/0.6

τ	angle-vel	Ax-Ay	angle-vel-fzl	Ax-Ay-fzl	angle-vel-Ax-Ay	angle-vel-Ax-Ay-fzl
1	0.8/0.6	0.8/0.6	0.6/0.6	0.6/0.5	0.6/0.6	0.6/0.7
5	0.6/0.4	0.8/0.6	0.4/0.4	0.4/0.4	0.5/0.5	0.4/0.3
10	0.4/0.5	0.7/0.6	0.3/0.3	0.3/0.4	0.5/0.4	0.3/0.3

Table A.6.: Comparison of the speed classification accuracies achieved by two limited predictors. The gray shaded values are the accuracies for a predictor limited to $10-20$ gait percent, whereas the other values are for a predictor for $60-70$ gait percent. For the input signals the abbreviations introduced in Table A.1 are used.

τ	angle		vel		Ax		Ay		fzl	
1	29	73	38	70	38	30	29	43	50	52
5	63	86	61	83	58	77	49	61	71	83
10	88	87	89	92	65	89	75	82	71	89

τ	angle-vel		Ax-Ay		angle-vel-fzl		Ax-Ay-fzl		angle-vel-Ax-Ay		angle-vel-Ax-Ay-fzl	
1	44	80	40	56	58	87	60	58	49	85	64	86
5	81	87	85	86	88	87	82	88	83	86	86	89
10	87	85	80	90	8	90	82	91	87	89	90	93