Multimodal interface for interactive cooperation with quadruped robots

Mikael Uimonen

School of Science

Thesis submitted for examination for the degree of Master of Science in Technology.
Helsinki 21.11.2022

Supervisor

Prof. Ivan Vujaklija

Advisor

M.Sc. (Tech.) Paul Kemppi

Aalto University
School of Science

Aalto University
School of Science

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author Mikael Uimonen

Title Multimodal interface for interactive cooperation with quadruped robots

Degree programme Master's Programme in Life Science Technologies

Major Complex systems **Code of major** SCI3060

Supervisor Prof. Ivan Vujaklija

Advisor M.Sc. (Tech.) Paul Kemppi

Date 21.11.2022 **Number of pages** 73 **Language** English

Abstract

A variety of approaches for hand gesture recognition have been proposed, where most interest has recently been directed towards different deep learning methods. The modalities, on which these approaches are based, most commonly range from different imaging sensors to inertial measurement units (IMU) and electromyography (EMG) sensors. EMG and IMUs allow detection of gestures without being affected by the line of sight or lighting conditions. The detection algorithms are fairly well established, but their application to real world use cases is limited, apart from prostheses and exoskeletons.

In this thesis, a multimodal interface for human robot interaction (HRI) is developed for quadruped robots. The interface is based on a combination of two detection algorithms; one for detecting gestures based on surface electromyography (sEMG) and IMU signals, and the other for detecting the operator using visible light and depth cameras. Multiple architectures for gesture detection are compared, where the best regression performance with offline multi-user data was achieved by a hybrid of a convolutional neural network (CNN) and a long short-term memory (LSTM), with a mean squared error (MSE) of $4.7 \cdot 10^{-3}$ in the normalised gestures. A person-following behaviour is implemented for a quadruped robot, which is controlled using the predefined gestures. The complete interface is evaluated online by one expert user two days after recording the last samples of the training data. The gesture detection system achieved an F-score of 0.95 for the gestures alone, and 0.90, when unrecognised attempts due to other technological aspects, such as disturbances in Bluetooth data transmission, are included. The system to reached online performance levels comparable to those reported for offline sessions and online sessions with real-time visual feedback. While the current interface was successfully deployed to the robot, further advances should be aimed at improving inter-subject performance and wireless communication reliability between the devices.

Keywords Human robot interaction, hand gesture recognition, surface electromyography, deep learning, neural networks

Aalto University
School of Science

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi
Diplomityön tiivistelmä

Tekijä Mikael Uimonen

Työn nimi Multimodaalinen käyttöliittymä interaktiivista yhteistyötä varten nelijalkaisten robottien kanssa

Koulutusohjelma Life Science Technologies -maisteriohjelma

Pääaine Complex systems **Pääaineen koodi** SCI3060

Työn valvoja Prof. Ivan Vujaklija

Työn ohjaaja DI Paul Kemppi

Päivämäärä 21.11.2022 **Sivumäärä** 73 **Kieli** Englanti

Tiivistelmä

Käden eleiden tunnistamiseksi on ehdotettu useita vaihtoehtoisia ratkaisuja, mutta tällä hetkellä tutkimus- ja kehitystyö on pääasiassa keskittynyt erilaisiin syvän oppimisen menetelmiin. Hyödynnetyt teknologiat vaihtelevat useimmiten kuvantavista antureista inertiamittausyksiköihin (inertial measurement unit, IMU) ja lihassähkökäyrää (electromyography, EMG) mittaaviin antureihin. EMG ja IMU:t mahdollistavat eleiden tunnistuksen riippumatta näköyhteydestä tai valaistusolosuhteista. Eleiden tunnistukseen käytettävät menetelmät ovat jo melko vakiintuneita, mutta niiden käyttökohteet ovat rajoittuneet lähinnä proteeseihin ja ulkoisiin tukirankoihin.

Tässä opinnäytetyössä kehitettiin useaa modaliteettia hyödyntävä käyttöliittymä ihmisen ja robotin vuorovaikutusta varten. Käyttöliittymä perustuu kahden menetelmän yhdistelmään, joista ensimmäinen vastaa eleiden tunnistuksesta pohjautuen ihon pinnalta mitattavaan EMG:hen ja IMU-signaaleihin, ja toinen käyttäjän tunnistuksesta näkyvän valon- ja syvyyskameroiden perusteella. Työssä vertaillaan useita eleiden tunnistuksen soveltuvia arkkitehtuureja, joista parhaan tuloksen usean käyttäjän opetusaineistolla saavutti konvoluutineuroverkon (convolutional neural network, CNN) ja pitkäkestoisen lyhytkestomuistin (long short-term memory, LSTM) yhdistelmäarkkitehtuuri. Normalisoitujen eleiden regression keskimääräinen neliöllinen virhe (mean squared error, MSE) oli tällä arkkitehtuurilla $4,7 \cdot 10^{-3}$. Eleitä hyödynnettiin robotille toteutetun henkilön seuraamistehtävän ohjaamisessa. Lopullinen käyttöliittymä arvioitiin yhdellä kokeneella koehenkilöllä kaksi päivää viimeisten eleiden mittaamisen jälkeen. Tällöin eleiden tunnistusjärjestelmä saavutti F-testiarvon 0,95, kun vain eleiden tunnistuksen kyvykkyys huomioitiin. Arvioitaessa koko järjestelmän toimivuutta saavutettiin F-testiarvo 0,90, jossa muun muassa Bluetooth-pohjainen tiedonsiirto heikensi tuloksia. Suoraan robottiin yhteydessä ollessaan, järjestelmän saavuttama eleiden tunnistuskyky vastasi laboratorioissa suoritettujen kokeiden suorituskykyä. Vaikka järjestelmän toiminta vahvistettiin onnistuneesti, tulee tutkimuksen jatkossa keskittyä etenkin ihmisten välisen yleistymisen parantamiseen, sekä langattoman tiedonsiirron ongelmien korjaamiseen.

Avainsanat Ihmisen ja robotin vuorovaikutus, käden eleiden tunnistus, lihassähkökäyrä, syväoppiminen, neuroverkot

Preface

I would like to thank all current and former VTTers who have had any part on helping me from getting this project started to finally getting it completed as well. Especially I would like to thank my advisor Paul Kemppi for his invaluable support with putting together this entity, which some could call just a robot with a piece of code. I would also like to thank my supervisor Professor Ivan Vujaklija for our great discussions and for always finding time for me. However, most of all I would like to thank my dear Elena, who has been a great source of support, inspiration and motivation throughout my studies.

Helsinki, 21.11.2022

Mikael Uimonen

Contents

Abbreviations

ADC	analog-to-digital converter
Bi-LSTM	bidirectional LSTM
BLE	bluetooth low energy
CNN	convolutional neural network
CWT	continuous wavelet transform
DoF	degrees of freedom
DWT	discrete wavelet transform
EEG	electroencephalography
EMG	electrocmyography
GAN	generative adversarial network
GN	gaussian noise
HD sEMG	high-density sEMG
HRI	human robot interaction
IMU	inertial measurement unit
LDA	linear discriminant analysis
LSTM	long short-term memory
LOOCV	leave-one-out cross-validation
mAP	mean average precision
MAV	mean absolute value
MAVS	mean absolute value slope
MFAP	muscle fibre action potential
MLP	multilayer perceptron
MNF	mean frequency
MU	motor unit
MUAP	motor unit action potential
NLP	natural language processing
PSD	power spectral density
RMS	root mean square
RNN	recurrent neural network
ROS	robot operating system
sEMG	surface electromyography
SVM	support vector machine
SW	sliding window
ViT	vision transformer
VT	visual transformer
WAMP	Willison amplitude
WL	waveform length
ZC	zero crossings

1 Introduction

Hand gestures, such as static and dynamic poses of the hands, are a natural way for people to express themselves, either in addition to spoken language or exclusively when using sign language. Using hand gestures has been shown to improve communication regarding topics such as motor actions [1] and in cases where hearing conditions are challenging [2] when compared to only using spoken language.

Hand gesture detection and classification based on sEMG have been studied extensively. The applied methods include various methods of traditional machine learning and artificial neural networks, where new methods are still being experimented with today. The most important unsolved problems with sEMG-based gesture detection are related to inter-session and inter-subject performance of the algorithms. That is, the accuracies of the algorithms degrade over time when switching the user or when donning and doffing the sEMG device without retraining, or at least recalibrating the system [3]. However, the degradation over time has been shown to be counteractable by providing real-time feedback to the user [4].

Traditional methods for sEMG signal analysis use predefined features, often called hand-crafted features, for extracting information from the signals. These features include mean absolute value (MAV) and root mean square (RMS) at the simplest, but overall, a multitude of various features and feature combinations have been suggested. The selected features can then be classified with various methods such as linear discriminant analysis (LDA) or support vector machines (SVM) [5, 6].

Neural networks, such as multilayer perceptrons (MLP) and LSTM networks, are additional options for the classification algorithm, which still require hand-crafted features for optimal performance. The features can be replaced with convolutional neural networks (CNN), which can use the raw, or slightly preprocessed signal as an input [7] and learn to extract features from scratch. In [8], however, the use of hand-crafted features improved the results of a CNN-based network, when compared to a similar network with raw signal-based input.

Physical devices, such as hand prostheses [9, 10] and orthoses, [11] and exoskeletons [12] have been controlled with sEMG-based gestures before. Re-enabling and improving existing human functionality this way can be seen as a natural application area for sEMG, as the devices are attached to the body and their activation can be built to mimic the activation of their biological counterparts.

The controlled devices are not limited to devices attached to human body mimicking the functionality of the body parts. Systems for human machine interaction [13] and sign language recognition [14] have been implemented with sEMG-based hand gesture detection. Gesture-based HRI interfaces using sEMG have also been implemented for mobile robots, such as in [15], [16] and [17], where the detected gestures were used to directly control the movement of the robots. Direct mapping of the gestures to movement can, however, require continuous input from the user, which increases the risk of muscle fatigue. In addition to being uncomfortable, muscle fatigue is known to affect the performance of sEMG-based control systems [18].

Other modalities than sEMG have also been used to implement gesture-based HRI interfaces for mobile robots. The modalities include camera based arm segmentation

[19], camera-based pose detection with [20] and without [21] depth information, and signals from a wearable IMU-device [22].

Mobile robots are usually either controlled with a hand-held remote controller, which can require constant input and the use of both hands from the person controlling the robot, or they are completely autonomous with limited interaction with the surrounding people. The task of a robot following a person has been studied for a long time with good results. While detecting and following a person are well established, the proposed solutions often lack the means of controlling or communicating with the robot. In addition to the hand-held controllers, voice and gesture-based controls have been proposed. While voice control has been applied with great success, it might not be viable in all circumstances, especially in loud environments.

The work described in this thesis aims develop a multimodal interface for HRI. More specifically, extended person-following behaviour for the robot is implemented, which can be controlled using hand gestures and the position of the user. The interface is based on an sEMG-armband (gForcePro+, OYMotion), which features an IMU in addition to sEMG sensors. The interface is used for controlling a quadruped mobile robot (Spot, Boston Dynamics) with the intention that it could be used in industrial environments, where the noise levels can be high. The visible light and depth cameras embedded to the robot are utilised as additional modalities for the interface.

First, an interface for the armband is developed for integrating it with the Robot Operating System (ROS), on which the HRI interface is also based on. The operating modes of the gForcePro+ armband are compared in a simulation based on a dataset recorded on another device. A dataset is collected of four subjects performing six types of hand gestures, which will be used to control the robot. The performance of a selection of neural networks is evaluated on this dataset to assess the feasibility of the chosen setup.

The interface for the robot is developed based on the gesture detection system. In addition to gesture detection, the interface features a visual person detection system, that is based on the YOLOv5 algorithm [23] and is used for detecting and locating the operator of the robot. The computational cost of the system is intended to be low enough to ensure deployability of the entire interface on a single-board computer (Jetson Xavier NX Developer Kit, NVIDIA), that is attached to the robot.

Finally, the gesture detection algorithm is validated with multiple users, as it is the most user-dependent component in the system. As a proof of concept, the complete interface is tested by one expert user.

2 Background

2.1 Measuring neuromuscular signals

Electromyography (EMG) is a technique for measuring the electric activity produced by muscles, whereas an electromyogram is a recording produced by using this technique. The electric activity can be measured noninvasively using sEMG, where surface electrodes are placed on the skin directly on top of the muscles whose signal is being measured. EMG can also be measured invasively by the means of intramuscular EMG and indwelling wire or needle electrodes [24, 25].

EMG measures the electrical potential caused by temporal imbalances of ions around muscle fibres. This imbalance originates from the nervous system in the form of action potentials travelling along motor neurons and results in the contraction of muscle fibres [24].

The axons of a motor neuron divide into small branches allowing the neuron to innervate multiple muscle fibres, which as a group are called a motor unit (MU). Motor neurons are connected to muscle fibres of an MU at the innervation zone. At the innervation zone, the action potential causes a series of chemical reactions which in turn open ion channels of the muscle fibre membrane. This results in the potential inside the cell rising from the resting potential of -90 mV to the peak depolarization value of 30-40 mV. Local transient changes of the potential inside a muscle fibre are called the muscle fibre action potential (MFAP) [24].

If the MFAP is large enough, there occurs a chain reaction of ion channels opening, which spreads the MFAP around the muscle fibre. When electrical potential of an MU, i.e., the individual MFAPs of the muscle fibres within the MU, is measured, the resulting compound potential is called motor unit action potential (MUAP) [24].

Furthermore, when EMG is used to measure the electrical potential changes in a muscle, single MUAPs are superimposed over each other and only this seemingly random signal is detected [24]. While reconstructing the individual MUAPs have been demonstrated, both using intramuscular EMG and sEMG [25–28], these methods are rarely used for gesture detection. Instead, the methods can be applied to determining the origins of neuromuscular diseases [26].

The amplitude of the superimposed MUAPs can be modelled as a random process that follows a certain probabilistic distribution. The maximum likelihood estimator of the signal depends on the chosen distribution; for a Gaussian distribution it would be RMS and for Laplacian distribution it would be MAV. The distribution of EMG has been shown experimentally to fall between these two distributions but fitting the Gaussian distribution the best on average. However, for amplitude estimation, MAV processing had a better signal-to-noise ratio compared to RMS processing [29].

The EMG signal can also be described as coloured noise, where the spectrum is characteristic to the current state and level of activation of the muscle. An example of the change seen in the power spectral density (PSD) of an sEMG signal is shown in Figure 1.

EMG signals can be recorded using devices that at minimum comprise electrodes, an amplifier, an analog-to-digital converter (ADC) and some applicable filters [30].

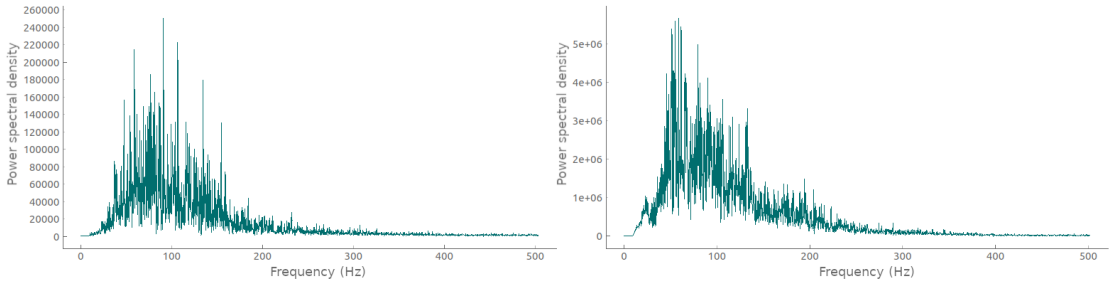

Figure 1: Left: PSD of an sEMG signal while the hand is stationary in a resting pose. Right: PSD from the same session when the hand is clenched into a fist. The sEMG signal was averaged over 8 channels recorded using a gForcePro+ sEMG armband with a sampling rate of 1 kHz and a window size of 4 s.

Usually either monopolar or bipolar configuration is used to measure EMG [24], but double differential configuration has also been used [31].

In bipolar configuration, two electrodes in addition to a ground electrode are used to measure the potential difference between the electrodes. An instrumentation amplifier, which is a type of differential amplifier, is typically used to perform the subtraction. Bipolar measurement is less susceptible to external noise and crosstalk from distant muscles when compared to its monopolar equivalent [30]. On the other hand, the advantage of monopolar measurement is that the actual shape of the MUAP can better be inferred from the resulting signal. This is because the bipolar configuration ideally measures the differential of the of the MUAP, which is furthermore convolved by the shape of the electrode [24].

The number of electrodes in a single device can be increased to form an array of electrodes, which can be used to measure high-density sEMG (HD sEMG). HD sEMG enables even better detection of crosstalk over single channel monopolar and bipolar methods [24]. The electrode count in HD sEMG varies per device, but arrays with more than 100 electrodes have been used [31].

The frequency content of the measured EMG is influenced by various factors. One of the most prominent of these factors is changes in MU conduction velocity. Increase in the velocity leads to higher frequency contents in the measured EMG. The opposite is typically observed during muscle fatigue; a good indicator of muscle fatigue is the power spectrum mean frequency shifting towards lower frequencies [24]. However, gesture classification accuracy in [4] did not decrease over time during a five-minute session when feedback was provided to the subjects. The subjects did report increased muscle fatigue, which implies that the experienced fatigue did not negatively affect the classification accuracy. Any changes in the frequency spectrum were not reported to further quantify the experienced fatigue [4]. Other attempts to counteract muscle fatigue include switching the model in real-time [18].

Other noticeable factors that change the frequency content of EMG include the MU recruitment pattern, which is generally believed to be effectively random during isometric contractions. That is, when the muscles are contracted without the length

of the muscle fibres changing, the MU activations are not synchronised. This changes during spontaneous and explosive movements, which leads to synchronised activation and increased contribution to lower frequency components [24].

The lower frequency components are also affected by movement artefacts, which are prominent during explosive and impact-like activities, e.g., sports. Physical activities can also change the placement of electrodes, which can alter the distance to the measured MUs. A larger distance narrows the spectrum and shifts it towards lower frequencies due to increased filtering effect of the surrounding tissue. Furthermore, the placement of the electrodes affects the quality of the measured EMG signal. The signal is weakened around the innervation zone, which is why the placement of the electrodes is often chosen specifically for each measured muscle [24].

Cold immersion has been shown to affect EMG signals. The RMS of the signal decreases when the muscles are cooled in addition to the amplitude of the signal increasing for sub-maximal exertions. However, the amplitude decreases for maximal exertions. [32].

Apart from the physiological and mechanical factors discussed previously, a number of parameters related to recording EMG can influence the quality of the signal and therefore also any inference that is based on the recorded signals. These parameters are discussed in Section 2.2 together with examples of the inference methods.

Higher-order spectra, such as bispectrum and trispectrum, can be used to study non-linearities in signals. In [33], bispectrum analysis of EMG signals from eight locations around the body was used to categorise 20 activities as either aggressive or non-aggressive. The sum of the estimated bispectrum magnitudes was used as an input for an extreme learning machine with a single hidden state. Regardless of the simplicity of the network, the algorithm was able to classify the actions with a good accuracy.

Neuromuscular electrical activity can also be quantified by using intraneural implants, which measure electroneurographic signals. The implants feature an array of electrodes, which are surgically placed directly on the nerves. As the subjects are usually amputees, motor intentions are performed instead of the corresponding gestures. The feasibility of using these implants to decode multiple motor intentions from the electroneurographic signals has been shown with temporal implants [34]. Severe challenges still remain before a long-term solution becomes available. The electrical signal weakens after time due to the foreign body reaction, which causes formation of fibrous tissue around the implant among other undesired side effects [35].

2.2 Myocontrol

Myocontrol, or myoelectric control, utilises measured EMG signals to control various devices such as prostheses [9, 10]. In practise, the inference for myocontrol must be performed in real-time, but both offline and online studies are included here for a broader overview. In addition to the control schemes, various parameters that can affect the performance of the system are discussed in this section.

The effect of sampling rate on EMG-based gesture classification was studied

in [36] by resampling the recorded signals. Here, ten hand and arm movements and a rest state were classified using 12 bipolar electrodes placed around the arm. The EMG signals were sampled at 1 kHz and band-pass filtered between 5 and 400 Hz. Five able-bodied subjects and two transradial amputees participated in the study. The best classification accuracy was achieved using the full 1 kHz sampling rate, 98.0% for the able-bodied subjects. When the signal was resampled at 500 Hz, the accuracy decreased only a little, to 97.2%. However, it should be noted that there is little room for improvement in the accuracy with sampling rates above 800 Hz, as the low-pass filter was set to 400 Hz. Below the 500 Hz mark, the classification accuracy decreased faster. At 300 Hz it was reported to be 95.7%, while 200 Hz and 100 Hz resulted in roughly 94% and 90%, respectively. The results of transradial amputees followed a similar curve from 75.9% at 1 kHz to 71.3% at 300 Hz. [36]

Sampling EMG signals at the Nyquist rate has been suggested to be too slow when waveform patterns like turns and spike amplitudes are being analysed. Instead, oversampling at a factor of 1.5 to 5 would be required. However, it was shown that oversampling is not required when gathering common timing and amplitude measures, such as onset time, burst duration and peak amplitude, and sampling at 1 kHz would be sufficient for these measures. On the other hand, undersampling at 500 Hz and 250 Hz started to have an impact on the results. The effects of undersampling were less severe on signals that were smoothed using a moving average filter. [37] Then again, applying a moving average filter acts as a low-pass filter on its own and applying it to the original signal reduces the required sampling rate, thus explaining the seemingly better results. Furthermore, the application of a moving average filter will result in losses of other important features, especially on the frequency domain. The act of undersampling and smoothing the signal was discouraged in the study [37].

Wearable sEMG armbands face different constraints than clinical devices, including ones related to physical size, cost, and ease of use. These limit the performance of sEMG armbands, which can be seen in the reduced channel counts, constrained electrode placement and maximum available sampling frequency. The effect of the reduced sampling rate on the classification accuracy of hand and finger movements was studied in [6]. The sampling rates used were 1 kHz, as it is often regarded as the Nyquist rate of the EMG signal, and 200 Hz, which is the sampling rate available to the Myo armband. The analysis was performed on multiple datasets available online. It is confirmed that sampling rate of 200 Hz is not sufficient to capture all the information available for sEMG measurements. While most of the energy of sEMG signals is focused on the range between 50 and 150 Hz, the spectrum of sEMG ranges up to 500 Hz and these higher frequencies should not be ignored [6]. Figure 2 illustrates the effect that reducing the sampling rate has on the zero crossings (ZC) feature.

Additionally, a system using the higher sampling rate benefited more from using additional features, when compared to a system with the lower sampling rate. It was concluded that features that work best for high sampling rates are not necessarily the best option for lower sampling rates. Finally, a multi-feature set consisting of L-scale, mean value of the square root, maximum fractal length, and Willison amplitude was proposed along with a larger feature set, which were both aimed to be used with the

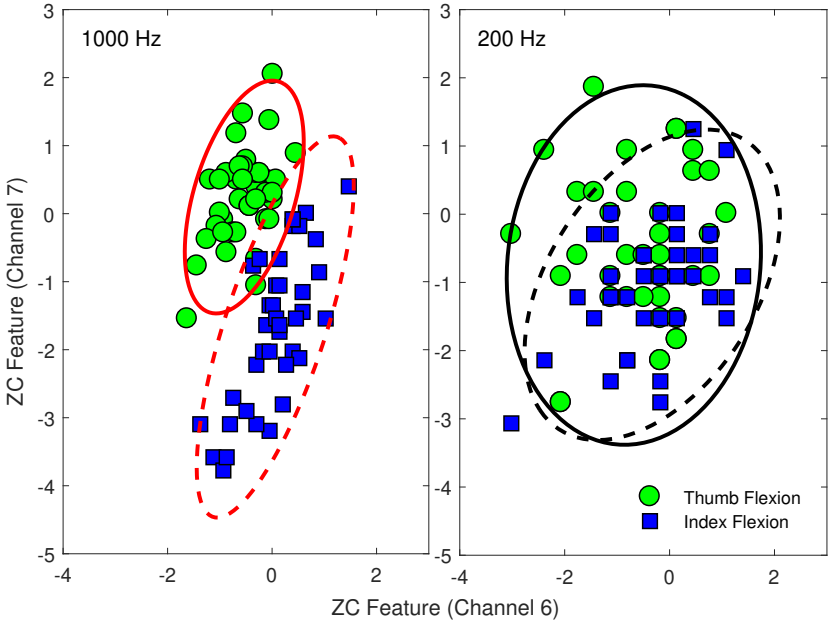

Figure 2: Reducing the sampling rate from 1000 Hz to 200 Hz results in less recognisable EMG patterns. ZC features extracted during thumb and index finger flexions are grouped closer together when using 200 Hz sampling rate. Adapted from [6].

lower sampling rate. [6]

When preprocessing sEMG signals, high-pass filters are usually set to relatively low values, between 10 and 50 Hz, whereas low-pass filters of 500 Hz are often used. In addition, a notch filter at 50 or 60 Hz is usually required to remove power line interference. [38] Another option is to use a higher value for the high-pass filter, so that the notch filter is not required. The use of a 120 Hz high-pass filter was reported to have statistically non-significant effect on classification accuracy when compared to a system with a notch filter and a 20 Hz high-pass filter [39].

Using multimodal data has been shown to improve EMG-based gesture recognition. CNN-based models, which will be further discussed in Section 2.3.1, trained on Ninapro DB2 and DB5 datasets achieved 83.7% and 90.0% recognition accuracies when using only sEMG signals as input. When simultaneously recorded IMU signals were also included in the model input, the recognition accuracies increased to 94.4% and 91.3%, respectively [40]. The raw accelerometer and gyroscope signals obtained from an IMU were stacked row-by-row to form an activity image [41], which was used as an additional input for the CNN-based models [40].

The gestures of the Ninapro datasets are relatively static postures. Even so, the previously described models were able to benefit from the inclusion of IMU data. The significance of an IMU in gesture recognition was found to be higher in [13], where the recognised gestures featured more dynamic hand movements, such as pull, push and multiple flicks in different directions. The authors used hidden Markov models and Gaussian mixture models to model the gesture classes and their observation probability distributions. When the gestures were recognised in

an inter-session intra-subject manner, the inclusion of IMU, in addition to sEMG, increased the recognition accuracy from 85.1% to 97.8%, whereas an accuracy of 92.8% was achieved when using only the IMU. The significance of an IMU was even more pronounced when the gestures were recognised in an inter-subject manner. Person-independent accuracy was 33.2% for sEMG and 70.2% for IMU only. When the modalities were combined, the inter-subject recognition accuracy was increased to 74.3% [13].

Recognising and translating sign languages to spoken languages has been proposed as one potential application for sEMG devices. A combination of sEMG and IMU signals was used in [14] to recognise 80 common signs of American Sign Language. These signs were recognised using hand-crafted features for both sEMG and IMU that were first selected from a larger set of features. The extracted features were processed using traditional machine learning methods, which include a naive Bayes classifier, a decision tree, a nearest neighbour classifier, and an SVM, which performed the best in this study. The SVM achieved a recognition accuracy of 92.3% when using only IMU signals in all cross validation, where the sessions and subjects were mixed. The accuracy increased to 96.2% when sEMG was also used. When both modalities were used in intra-subject inter-session manner, the accuracy decreased to 85.2%. This result shows how the system would perform in a scenario, where a classifier is trained for one person and later used by the same person in a separate session. It was concluded that the proposed system was not capable of inter-subject recognition, as the obtained accuracy for SVM was below 40% [14].

When gesture detection is performed in real-time, with a human in the loop, the input delay of the system should stay within acceptable limits. This limit has been suggested to be roughly 300 ms [42], which sets an upper boundary for the available processing time. Often, EMG signals need to be windowed before any further analysis can be made. While a longer window includes more information, which has been shown to increase classification accuracy [43] it can also increase the delay before a gesture is recognised. For these reasons, window lengths between 100 and 500 ms are usually chosen [38, 40, 43]. In [43], the amount of overlap in the window had no effect on the classification accuracy, if the window length stayed constant, when tested with handcrafted features and LDA for classification. Instead, window overlap affects the update frequency of the system [43].

As can be expected, increasing the number of EMG channels has been shown to improve the classification accuracy. In addition, the accuracy improvement from using a longer window depends on the channel count; it is more beneficial to increase the window length when only a few channels are used. However, increasing the channel count had adverse effects in a few cases. The chosen locations of the EMG electrodes were changed at the same time, which could have an effect to the results [43].

One of the key challenges with sEMG-based control schemes is achieving long-term reliability while maintaining rich functionality. sEMG signals can become unreliable due to transient changes, such as electromagnetic interference, skin perspiration, electrode shift and fatigue. [44] To achieve robust long-term control over a robot arm with seven degrees of freedom (DoF), the authors of [45] used an HD sEMG with 192 channels and 2048 Hz sampling frequency to create a control scheme inspired by

muscle synergies. The scheme is calibrated using HD sEMG recordings of $n = 16$ distinct motions of the hand, where a pair of motions correspond to a single DoF, such as flexion and extension of the index finger. A channel weighting matrix $\mathbf{W}$ is generated from the recordings using a DoF-wise nonnegative matrix factorisation, as explained in [46]. The matrix $\mathbf{W}$ is then used to extract $\hat{\mathbf{W}}$, by thresholding the elements of $\mathbf{W}$, applying a Gaussian blur for each reshaped column, and merging similar columns based on cosine similarity until $k < n$ columns remain. In this case, $k = 4$, which is the number of quasi-independent control inputs $\hat{\mathbf{F}}(t) = \hat{\mathbf{W}}^T\mathbf{Y}(t)$. $\mathbf{Y}(t)$ contains the linear envelopes of the sEMG observations. Extracting the activation signals this way is claimed to reduce influence of crosstalk between the activations. The control scheme is shown to be stable over multiple days and the subjects improved their performance over the course of the three sessions included in the study [45]. However, the 192 signals acquired using three 8×8 electrode grids may not always be available, especially outside of laboratory conditions.

Two-dimensional simultaneous and proportional control was achieved using an autoencoder (AEN). Here, 16 channel monopolar sEMG was recorded at 2048 Hz and segmented using 100 ms non-overlapping windows. RMS was calculated for each channel in a given segment and the RMS values were used as an input to the AEN. The encoder of the AEN was used to project the values to a two-dimensional representation, corresponding to the two dimensions of the motion of a human wrist, namely flexion/extension and radial/ulnar deviation. The results were validated in an online experiment, where the AEN-based method outperformed a model using direct sequential control, commonly used in commercial prostheses. The proposed AEN featured a single layer in each the encoder and the decoder. The authors note that even though the assumption of linear mapping from RMS to the kinematics might not be ideal, it can be compensated by the feedback loop introduced with online control. An advantage of using an AEN is that it supports unsupervised training, which can save expenses and reduce the active training time for subjects. [9]

2.3 Neural networks

2.3.1 Convolutional Neural Networks

Convolution, or discrete convolution in particular, is a mathematical operation of applying a certain pattern, a kernel or a filter, over a given data sequence. It results in a new instance of that data, which has been altered by the operation. This is similar to windowing and calculating feautures in the resulting samples, which was discussed in chapter 2.1. When windowing or pure convolution is applied, the applied kernels are usually handcrafted by the practitioner and significant effort is required to find the optimal kernels.

CNNs are neural networks that include one or more convolutional layers, but they usually have other layer types as well, such as fully connected and max-pooling layers. Convolutional layers apply the convolutional operation over the input data using kernels that are learned automatically during training, without the need of handcrafting the kernels. Convolutional layers provide locality and translation

invariance [47], which is their main difference when compared to fully connected layers.

CNNs are most commonly applied to data that is inherently two-dimensional, such as images from various sources. These sources include, among others, photography, medical imaging, range imaging and spectrograms, that have been calculated from 1D signals. For these kinds of data, 2D CNNs are a natural choice, which use 2D kernels to process the data.

Even though 1D signals can be transformed to 2D, either by calculating the spectrogram or by stacking multiple signals or parts of the same signal row by row to form a 2D grid, or a signal image, it is not always desirable. Applying a 2D CNN to a 2D grid of stacked signals, enforces local relations to the signals, where there might be none. In case the CNN learns kernels that only focus on certain signal sources at a time, i.e., only one row has non-zero elements, it wastes computational resources. Furthermore, a kernel of specific size fails to capture patterns between signals that are not stacked close enough for the kernel. To circumvent this, a study in human activity recognition found that stacking the signals multiple times in different orders increased the classification accuracy [41], which could imply that a 2D convolution would otherwise fail to capture the necessary relations.

Nevertheless, signal images have become popular in sEMG-based gesture recognition studies. In [40], multiple feature sets were used to construct corresponding signal images, which were then used as inputs for a multi-view CNN. Another model, a hybrid between a CNN and a recurrent neural network (RNN) was proposed in [8], where feature-based signal images were used to improve classification accuracy. RNNs are discussed in the next section. It was also suggested that using handcrafted features instead of raw signals improves the accuracy, even when using CNNs.

In [4], multiple CNN achitectures were proposed with different modalities derived from sEMG signals, which included raw sEMG signals, spectrograms and continuous wavelet transforms (CWT). Similar to [8], the different modalities were treated as an image and 2D CNNs were applied. The proposed models were tested on two datasets; the Myo dataset, on which CWT performed the best, and the Ninapro DB5 dataset, on which raw sEMG performed the best. In all cases, the inclusion of transfer learning improved the results, as did increasing the size of the training data. However, no upper limit for the sample sizes were seemingly reached. The authors reported an offline accuracy of 98%, whereas the average online accuracy with feedback was between 92% and 95%, as approximated from a provided graph. The lowest accuracy from an individual session was 84%, whereas the lowest individual accuracy without feedback was as low as 51% [4].

Instead of forcing 1D signals into a 2D shape in order to process them using CNNs, the 2D convolutional layers can be replaced using their 1D counterparts. This was demonstrated in [48], where 1D CNNs were used to detect particular heartbeats from an electrocardiogram. As noted earlier, convolutions resemble window-based feature calculation, that has been a common practice in signal processing for a long time, but the kernels of CNNs are automatically learned. This way, feature extraction and classification are unified into a single process.

2.3.2 Recurrent Neural Networks

A review of popular RNN cell structures and architectures was presented in [49], which is recommended for more details. The recurrent cell, used in standard RNNs, is the simplest of the presented cell structures, which consists of only the input and output connections, and the connections for the previous hidden state h_{t-1} and the output of the current hidden state h_t. The hidden state, which also serves as the output of the cell at time t, can be updated as follows:

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b), \tag{1}$$

where W_h and W_x denote the weights for the previous state h_{t-1} and the input x_t, b is the bias and σ stands for the sigmoid function.

A common problem with many ANNs is the problem of vanishing gradients and RNNs are no different in this sense. This is especially true when trying to learn long-term dependencies. LSTMs improve the performance of RNNs by introducing gate-like structures and cell state, which together enable a better flow of information between time steps. The gates consist of a fully connected layer wrapped into a sigmoid function, outputting a value between 0 and 1. This multiplied with the gated signal allows a better control over the flow of information.

Many modifications have been proposed to the LSTM and its cell structure, one of the most important ones being the forget gate. The original LSTM contained only input and output gates, but the addition of the forget gate allows the network to selectively discard some of the previous information in the cell state. The forget gate has been proven to improve results, and it is usually considered to be part of the standard LSTM cell.

The LSTM cell can be used alone, but they can also be combined in various ways for more demanding applications. The most straightforward way is to stack LSTM cells together in layers to form a deeper network. The outputs of a given layer then act as inputs for the next, while the output and the cell state of layer L at time t are also forwarded to the next iteration of the same layer at time $t + 1$. Gated-feedback LSTMs on the other hand have connections between all layers across the time dimension, not only between layers of the same depth.

The recurrent property of LSTMs (and other RNNs alike) makes them a good choice for online time series analysis. That is, the input data of the time series can be processed one sample or a small batch of samples at a time. Given a sufficiently powerful computer, this can be done at the same rate as the samples are acquired. Increasing the batch size can reduce the inference time required. It needs to be noted that other types of ANNs, such as MLPs and CNNs, can be used for online time series analysis as well, but this usually requires larger batches to compensate the lack of memory in the models. The increased batch size can result in some redundant computation if the input batches are overlapping.

Online classification of hand gestures using LSTMs based on sEMG signals was studied in [30]. Here, the classification was based on a single feature, standard deviation, that was calculated for each channel separately using a moving window with a size of 100 samples (0.5 seconds with a sample rate of 200 Hz). The LSTM

implementation were compared against a MLP, a basic RNN and a GRU. The recurrent networks received the extracted feature one time step at a time, while the MLP received batches of 200 or 300 samples, with a step of one. As a consequence, the training and inference times of the MLP are substantially longer than those of the GRU and LSTM. Nevertheless, the authors reported a high single frame classification accuracy ($\geq 95\%$) and gesture detection accuracy ($\geq 98\%$) for all of the tested models when trained on UC2018 DualMyo Hand Gesture Dataset, which contains 7 gestures and a resting state. However, no online experiments with humans in the loop were conducted [30].

One study [8] used a hybrid CNN-RNN architecture with attention mechanism to recognise hand gestures from sEMG-signals alone. The authors used already available datasets, such as DB1 and DB2 from the Ninapro database [50], which are recorded using 10 separate electrodes. Here, the effect of image representation methods was studied between seven different methods. These include simply stacking the channels and frames to form an image, rearranging the channels into multiple concurrent configurations, and performing a two-dimensional FFT on the result of the previous method. However, the authors concluded that the best approach is to use the classical Phinyomark feature set [5] and rearrange the components of the feature set instead of the original signal as in the other methods. The resulting image representations were used as an input to the neural network. The use of a feature-based image improved the classification accuracy by several percent, whereas the addition of the attention mechanism improved the results by a few tenths of a percent. The authors report a classification accuracy of 87.0% and 82.2% for DB1 and DB2, respectively [8].

A hybrid model consisting of CNN backbone and a Bidirectional LSTM (Bi-LSTM) was used in [7] to classify hand gestures from sEMG signals. The CNN layers were responsible for learning to extract low-level features from the signal. As CNNs are better at extracting spatial features rather than temporal structures, the CNN block was followed by a Bi-LSTM before the final fully connected layers. With Bi-LSTMs, the input sequence is processed both in the natural direction, the same as with a regular LSTM, and in a reversed direction. This is done using two separate models whose outputs are later fused together to give the combined results. In this work, two stacked Bi-LSTM layers achieved the overall best result. The hybrid model was tested using various datasets, including Ninapro DB2, which the model was able to classify with 95.9% accuracy using a window size of 200 ms. [7] One advantage of regular LSTMs is their ability of analysing the input signal one window at a time as soon as the windows become available. Using a Bi-LSTM loses this advantage as the final window needs to be known when starting the process, which introduces delay in the output. Alternatively, all the windows could be processed every time a new window becomes available, which instead increases the computational cost.

2.3.3 Transformers

Transformers are a relatively new group of architectures designed to utilise self-attention, a way to control the amount of influence each component of the input

has on the inference, replacing operations such as convolutions or recurrent patterns seen in other types of models. The original Transformer [51] was introduced in 2017 for machine translation, but it has since been adapted for various other tasks. They have been reported to outperform LSTM networks in natural language processing (NLP) tasks, such as speech translation and text-to-speech. [52] Outside the realm of NLP, transformers have been applied to tasks such as image classification [53, 54], object detection [55], heartbeat classification for arrhythmia detection [56], and hand gesture classification from depth images [57] and from sEMG signals [58] as well.

One study used a transformer-based architecture to recognise hand gestures from multimodal data, featuring depth, visible light and IR sensors. The data from the depth camera was used to estimate the surface normals, which were used as an additional modality. The data was made compatible for the architecture by using a CNN to extract features from the images before feeding the output to the transformer. The final classification was performed using a fully connected layer. The network was evaluated against two datasets with 25 and 12 different gestures and 12 and 40 subjects, respectively. The authors report classification accuracies of 87.6% and 97.2% for these datasets. [57]

The attention function is based on transformations of the input called queries, keys and values, which are inspired by search engines and recommender systems. Conceptually, a query is compared to the keys and the resulting scores are used to scale the values, thus giving a different result for each query. [59] The particular attention function of the Transformer, called scaled dot-product attention, is given in equation 2, where the dot-product between queries and keys is further scaled by the square root of the keys dimension to allow a better flow of gradients.

$$Z = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{2}$$

where Z equals the resulting attention matrix, Q, K and V denote the query, key and value matricies and d_k is the dimension of the keys. In addition to dot-product attention, different forms of attention such as additive attention has also been used. [51]

Instead of applying the attention only once for each query, the Transformer uses multiple attention heads to attend to information from different representation subspaces. That is, learned linear projections are used to project the keys, queries and values to be used as input for the attention heads. The final multi-head attention is acquired by concatenating the output of the individual heads and projecting the result once more. [51]

Traditional RNNs do not feature an attention mechanism, although attempts at including attention to RNNs have been proposed, such as [8]. As mentioned in section 2.3.2, RNNs can only use the current input and the stored state, which represents the memory of the model, to determine the corresponding output. However, this can conceptually be seen as a form of attention, which is predetermined by the architecture. Such predetermination is called the inductive bias of the model, which is the set of choices made to help the model learn the underlying patterns of the data. RNNs are hard coded to look only into present and the past with a decaying

memory.

Unlike RNNs, Transformers take the input as a whole. With the help of self-attention, they are able to focus on specific elements of the input. These elements can both precede and succeed the current input element, with arbitrary distances within the input dimension. This ability to choose where to pay attention to is learned from the training data and thus the model is less restricted; it has a smaller inductive bias. The disadvantage of the smaller inductive bias and a larger representational power is that Transformers generally require more training data to achieve superior results to RNNs. [60]

Large datasets required to train a Transformer in a supervised manner can be hard, expensive and time consuming to acquire, as it would require manually labelling all the samples. Instead, Transformers can be pre-trained on a large unlabelled dataset using methods of self-supervision, such as predicting masked parts of an input sample based on the rest of the sample or predicting whether a given sample is a successor to another sample. After pre-training, the models need to be fine-tuned for the specific task. Fine-tuning is a form of transfer learning, where the model is transferred to a down-stream task. This is done by replacing the output layers of the model and training all the parameters end-to-end. [61]

As transformers were aimed specifically for NLP tasks, retargeting them for image processing is not trivial. The rest of this subsection present two different approaches for dealing with this problem and try to answer why this would be desirable in the first place.

When dealing with image-like data, the convolution operation used in CNNs has proven to be successful and before transformers, convolutions had no clear competition or noteworthy alternatives. This is due to the inherent idea of convolution, that the involved operations should be applied equally to all the parts of an image. Thus, the weights of the convolution layers need to be learned only once and the structures and the shapes present in an image will be treated the same, regardless of their location. The convolutional layers are especially good when extracting low-level features, such as corners and edges, but when convolutional layers are used for detecting higher-level features, they can turn out to be redundant when most of the learned high-level features are not present in every image. [54] It has been shown that self-attention heads are capable of expressing any convolutional layers, potentially replacing them in the future. [62]

Visual Transformers (VT) were introduced to overcome the redundancy of convolutional layers when detecting high-level features. Furthermore, VTs are argued to improve on two other aspects of convolutional layers. First, convolutional layers treat every pixel similarly, regardless of their importance. Second, convolutional layers operate on small regions, making the networks struggle to relate spatially distant concepts. The attention mechanism of transformers addresses both of these problems by choosing the weights depending on the importance of the areas [54].

To adapt transformers for visual tasks, low-level features are extracted using convolutional layers. Next, visual tokens are extracted from the feature set using a recurrent tokeniser, which incrementally refines the set of visual tokens. The tokens of the previous layer are utilised when extracting the tokens of the next layer. This

can be formulated as

$$\begin{aligned} W_R &= T_{in} W_{T \to R}, \\ T &= \text{softmax}_{HW}(X W_R)^T X, \end{aligned} \tag{3}$$

where T_{in} and T denote the tokens of the previous and the current layer, X denotes the input consisting of the feature maps, and W_R and $W_{T \to R}$ denote the weights of the tokeniser. As the weights W_R depend on the previous tokens, the first set of tokens are calculated without the recurrent tokeniser, using a fixed set of learned weights W_A instead of W_R. The resulting visual tokens are then used as an input for a transformer [54].

The authors evaluate the model in a classification task with the ImageNet dataset but also in a semantic segmentation task with two datasets. The VTs used for classification are based on different versions of the ResNet architecture, where the last stage of convolutions is replaced with VT modules. This improved the Top-1 accuracy by 2.2 and 1.7 points when using ResNet18 and ResNet34 as the base model, respectively, while decreasing the required floating-point operations. Furthermore, when the training procedure is fine-tuned using stronger data augmentation, stronger regularisation, knowledge distillation and other strategies, the improvement in Top-1 accuracy increased to 3.0 and 2.2 points when compared to similarly trained base model [54].

Vision Transformer (ViT) is another approach at adopting Transformer architecture for visual tasks. ViT stands out from previous attempts by replacing convolutional layers completely with a Transformer encoder, but they also experiment with hybrid models between CNNs and ViTs, similar to VTs. In order to process image data as sequences, the images are split into patches of given size, such as 16×16 pixels. These patches are linearly projected and a position embedding is added before using them as a linear sequence of tokens for the encoder. The position embeddings are added to incorporate some information about the spatial structure, which would otherwise be lost [53].

As with other Transformers, ViT also requires significant amount of pre-training to be competitive against other architectures. Similar to training transformers for NLP tasks, self-supervision can also be utilised for ViTs. Here, masked patch prediction was studied in a preliminary manner, where the model learns to predict the contents of a masked area, based on the rest of the image. While this approach improved the results when compared to training from a scratch, the authors focused more on supervised pre-training, as it yielded better results [53].

Both pure and hybrid ViTs were compared to other models, such as slightly modified ResNets using datasets with varying sizes. When pre-trained with Google's internal JFT-300M dataset with 300 million labelled images, the model achieves 88.6% in ImageNet classification task. The model was pre-trained for 6.85 TPUv3-core-years, meaning it would take 10.3 months to train the network on a single tensor processing unit with 4x2 cores [53].

On smaller datasets, the hybrid models outperformed the pure ViTs. It was proposed that the inductive bias, the set of assumption embedded into the model

structure, originating from the convolutional operations are the key for better learning when the data is more limited. With huge datasets, such as JFT-300M, the ViTs were able to learn the necessary structure from a scratch to overtake the hybrid and purely convolutional models. Further experiments showed that the attention of pure ViTs in the first layers is focused on smaller areas than that of the hybrid model, as it needs to concentrate on local features too [53].

ViT has been successfully applied to detecting hand poses from sEMG data. In [58], a ViT was trained on NinaPro DB2 dataset, which includes 12 channel sEMG signals sampled at 2 kHz of 40 subjects performing 49 different hand movements. However, only 17 of these movements were included in this study. To be able to feed continuous signals to the Transformer encoder, the input signals were segmented to fixed-size patches with no overlap. A couple of model variants were compared in the study, where the base model featured one layer, a model dimension of 32, and MLP size of 128. These models were evaluated for window sizes of 200 and 300 ms. The best variant of the tested models featured one layer, a model dimension of 64 and MLP size of 256. This applied to both window sizes, while 300 ms window resulted in a slightly better classification accuracy of 82.9%. [58] In comparison, a CNN-RNN model with attention resulted in 82.2% classification accuracy in the same dataset, while classifying all 49 gestures included in the dataset. [8] The similar results could imply that either the training data is not sufficient to fully harness the representational power of Transformers, or that the data acquired by the sEMG sensors does not contain sufficient information to distinguish between all the 49 gestures included in the dataset. As the dataset is fully annotated and no other datasets were used, pre-training and fine-tuning were not used either, which could be used to improve the results further. [53, 61]

The robustness and accuracy of a machine learning solution can be pushed further by ensembling, i.e., combining multiple models for the same task. The combined models should be sufficiently different from each other, to bring enough diversity to the ensemble, trained separately, and they should all be well-performing. The importance of the last criterion can be reduced by weighting the models differently based on the validation data. [59]

Model ensembles have been popular in machine learning competitions [59], but combining multiple models is not always practical, especially in real-time applications. This is because running an ensemble can be computationally expensive, while increasing the size of the ensemble yields diminishing returns. Furthermore, the GPU-time required for training the ensemble might become too large. [63]

Knowledge distillation, also known as model distillation, is the act of training a smaller model based on the outputs of a larger one. In a classification task, the models typically produce a probability distribution of the target classes by using a softmax activation as follows

$$q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}, \tag{4}$$

where q_i and z_i are the probability and the logit of the class i, respectively, and T is the temperature, which is usually set to 1. Increasing the value of T produces

softer probability distribution for the classes, i.e., the differences between the correct and incorrect classes are smaller. This is the key for model distillation, as these probabilities can be used as soft set of targets when training the distilled model. These soft targets convey more information about the relationships between the classes than the usual hard targets. Model distillation has been proven to improve the results of the smaller model, when comparing to a similar model trained only on the raw data. Additional benefit of using model distillation is that a smaller dataset may also be used to speed up the training of the distilled model. [63]

Model distillation was used to improve the results of a VT, as mentioned previously. Here, an automatically designed neural network was used as a teacher to demonstrate the improved learning capacity of the proposed model. [54]

2.3.4 Data augmentation

Data augmentation is an essential technique in deep learning, where the already existing training data is artificially inflated using various methods. The premise with data augmentation is that the information relevant to the task, that is included in the training data, can be better extracted when using these methods while avoiding learning patterns that are only specific to the used data set, i.e., overfitting. Data augmentation has proven to be effective in reducing overfitting and it is often used in tasks that use image data as input. The applied methods include geometric transformations, such as rotating and scaling the data, addition of noise, such as Gaussian noise and irrelevant features, and using Generative Adversarial Networks (GANs) to generate new samples similar to the existing ones. [64]

All the data augmentation methods used for image data are not directly applicable to biosignals, such as EMG and electroencephalogram (EEG), even though a time-series of these signals can be structured like an image. Especially the notion of geometric transformations is not sensible in the context of an array of 1D signals. The location of a biosignal source has a heightened importance when compared to a location of a given feature in an image. For example, it matters more from which muscle an EMG signal originates from when classifying hand gestures, than it matters where exactly an eye is located in an image when classifying images of animals. Thus, reordering the signals is not always feasible, but great care must also be taken when applying other transformations. A human can easily tell if a transformed image still resembles the original, but it is not as easy to tell whether a transformed biosignal still possesses the characteristics relevant to the classification of that signal. [64]

Data augmentation in biosignals has been gaining attention since 2015, although many of the techniques used for augmenting biosignals have been in use for much longer. A review [64] of data augmentation methods published in 2020, lists seven different types of data augmentation methods that have been applied to augmenting EEG signals. From the listed augmentation methods, sliding windows (SW), GANs, sampling, and noise addition were the most popular among the referenced studies. Here, sliding windows refers to slicing the signal in multiple possibly overlapping segments. Sampling refers to over- and subsampling the items in the training

set, which is usually used to address imbalanced datasets, a problem where some classes in a dataset are severely underrepresented. Oversampling was used in [65] to prevent the class imbalance problem when classifying sleep stages based on EEG. More specifically, a Synthetic Minority Oversampling Technique was used, which interpolates between two nearby datapoints to create new similar instances. This is only applied to values extracted using hand-crafted features, such as PSD and spectral entropy. To augment the raw signals, different temporal offsets and noise addition were used.

Signal shifting and scaling was applied to EEG signals in [66] in order to augment the data used to train an MLP with two hidden layers to classify the intrinsic motivation of the subjects when pressing a button. Additionally, the authors experimented with restricting the scaling and shifting to only a certain segment around the peak of the signal. This was argued to be effective as it would emulate differences in mental states, rather than shifting and scaling the signal as a whole. The authors report a 8.3 % improvement for a dataset containing 20 signal samples of 10 participants each. Increasing the number of training samples per participant to 40 and 60 diminished the effect of data augmentation. The authors note that the applied methods improve training accuracy especially for smaller datasets. [66] However, larger networks that would benefit more from larger data sets and data augmentation were not considered in this study. Furthermore, the scaling and shifting was limited to specific integer values, such as ± 5 % and ± 5 ms, rather than ranging between these values, which reduces the amount of possible variation in the augmented data.

For sEMG signals (datasets from NinaPro-DB1 [50] and putEMG [67]), various data augmentation methods were experimented with in [68], including the most common ones: Gaussian noise and sliding windows. Instead of adding Gaussian noise uniformly across the signals, the variance of the noise was made relative to the signal amplitude, i.e., the noise was added with a constant signal-to-noise ratio. Other, less common, methods include magnitude warping, in which the amplitude of the signal is warped using an elementwise product with a randomly generated cubic spline with a mean of 1, and wavelet decomposition, in which a signal is transformed using the Discrete Wavelet Transform (DWT), the extracted detail coefficients are modified and the augmented signal is obtained through the Inverse DWT. Simulation of sEMG signals was also experimented with, but neither of the simulation methods tried yielded results comparable to the other methods. Both of the presented simulation methods were based on an idea of first estimating the overall shape of the original signal and then augmenting it with artificial features [68].

Domain adaptation was used for correcting domain shift related to inter-session and inter-subject changes when classifying hand gestures from sEMG signals. The proposed architecture consists of a single fully connected layer responsible for applying the domain adaptation to the sEMG signal before feeding it to an LSTM. This approach was evaluated using both sparse and HD sEMG, and while the approach did not reach the state of the art in intra-session scenarios of the time, the results did exceed the state of the art in inter-session and inter-subject scenarios, proving the benefit of domain adaptation when the conditions are changing. [69]

2.4 Hand gesture detection and interpretation

In this work, hand gestures are categorised into discrete and continuous gestures, where discrete gestures are further categorised into static gestures (or postures) and dynamic gestures. Examples of static gestures include the gestures from the game Rock Paper Scissors, or the alphabet of the American Sign Language (ASL). Swipes, flicks and waves, and various words in ASL or other sign languages include movement of one or both hands and are thus categorised as dynamic gestures. Continuous gestures, such as indicating the size of an object with two hands or showing an angle by bending a wrist or an elbow are static in a sense that a single snapshot of the gesture would be enough to convey the expressed information at that time. On the other hand, these gestures can be maintained for longer durations, and they can become more dynamic if the encoded information within the gesture is varied with time.

It has been debated whether gestures convey meaningful information to other people when accompanied with the related spoken language. While some speaking situations and topics of discussion do not usually require hand gestures to convey the message, including gestures have been shown to be beneficial when the gestures depict motor actions or when speaking to children [1]. Furthermore, it has been shown that when the hearing conditions are challenging, regardless of whether the challenge originates internally from a hearing impairment or externally from the environment, included gestures influence the perceived message [2]. That is, under noisy circumstances, gestures can be used to improve the understandability of conveyed messages. Finally, communicating abstract shapes is improved when hand gestures are allowed in addition to spoken language. There are cultural differences, however, as it was found that native Italian speakers benefit more from using gestures than native English speakers [70].

Hand gestures are generally recognised using either remote sensing devices, such as cameras and depth sensors, or wearables, such as data gloves and IMU and sEMG armbands, both the modalities are often combined with sensor fusion techniques to improve the performance of the system.

One of the first viable solutions for image-based hand gesture classification is presented in [71], where a CNN was used to classify the number of fingers shown. The system achieved the highest classification accuracy of the time, but when compared to modern solutions it has several disadvantages. The system is based on colour segmentation, which is dependent on the current lighting conditions. To facilitate the segmentation, the user is required to wear a coloured glove. Finally, the images are cropped in advance to include only the hand in question. These limitations inhibit the use of the system in most of the real-world scenarios, but the work acts as a first step towards systems capable of working in such scenarios.

Using gloves as an aid is a conceptually straightforward way to implement a solution for hand gesture recognition. Coloured gloves have been used with RGB cameras to help separating hands from the background. In [72], a glove with coloured fingers and palm was used, to directly extract finger locations using colour-based segmentation. Using the distances from finger centroids to the palm centroid and

the angles between the fingers as the features, a learning vector quantisation model was trained to classify 13 static gestures. The gestures were performed towards a webcam, at an unspecified distance. The best model achieved a 97.8% classification accuracy. [72]

The coloured glove and the camera were replaced with a data glove in [73]. The output of 15 sensors of the data glove, that measure the bending angles of fingers, were used as an input for a small MLP, which classified the hand shapes. Here, the task was considerably harder than that of the previous one: to recognise 50 words of American Sign Language. In addition to hand shapes, the system incorporated other metrics from the data glove as well, that describe hand movement and orientation relative to the user. The hand shape classifier reached an accuracy of 98% and when it was combined with the other metrics to form a recognition model for ASL words, the recognition accuracy was about 90%. [73]

RGB images can contain great amount of information, with varying levels of relevance when it comes to hand gesture recognition. That is, every detail included in an image does not necessarily convey information about a particular gesture being performed. Excessive background clutter is a common problem for vision-based algorithms, such as ANNs. This is why different preprocessing steps and additional modalities have been experimented in an attempt to improve accuracy and robustness of the models. In [74], RGB images are supplemented with semantic segmentation masks of the hands in the images. A lightweight segmentation method is shown to be faster than also used optical flow, while the resulting model achieves comparable or even better accuracies. The used dataset included 13 classes performed by 50 subjects, with a 74.8% average accuracy. It is worth noting that when double tap gestures were removed from the dataset, the average accuracy increased to 85.1%, as the repetitive gestures were not confused with single tap gestures. [74]

Traditional computer (von Neumann architecture / GPU) and neuromorphic computing-based approaches to hand gesture recognition with ANNs were compared in [75]. In a similar manner to ANNs, neuromorphic computing is also inspired by biological neurons. Where ANNs are mainly based on matrices and matrix operations defined in software, neuromorphic computing features hardware-level network of spiking neurons, which enables event-based design and fine-grained parallelism similar to their biological counterparts. Low power consumption and low latency are some of the key features of neuromorphic computing, which make the technology a potential enabler of future prosthesis control and autonomous systems. [76]

The five hand-gestures in [75] were observed using the Myo sEMG armband and an event-based camera. MLPs and CNNs were trained for both modalities separately and when fused together, with a GPU and neuromorphic chips as computational devices. Even though the models were kept similar between the devices, the GPU achieved better results in terms of accuracy when the stimulus durations were small (< 100 ms) and neuromorphic chips reached only a similar level with longer durations. When using just EMG as the input, the GPU model was consistently better across all stimulus durations, although the accuracy was lower for both devices when compared to models with camera-data and sensor fusion inputs. Two features (MAV and RMS) were calculated from the EMG and used as the input. The accuracies with only

event-based camera as input were slightly lower than with both modalities, but still higher than EMG alone. Even though GPU-based models generally reached better accuracies with smaller inference times, their energy consumption was considerably higher. [75]

Leap Motion Controller [77], which is an embedded device for hand tracking, has extensively been used for recognising the signs of a sign language [78–80] and other gesture detection tasks, such as the choices in the game rock-paper-scissors in [81]. The results for sign language recognition, as listed in [80], range from below 80% to over 97%. The described tasks involve recognition of various mixtures of the 26 letters in the alphabet of the American Sign Language, the ten digits, and some sample of the vocabulary. Overall, the recognition accuracy depends heavily on the number of classes involved. In [81], three gestures were recognised with an accuracy of 98% by using a three-layered MLP, whereas in [79], a deep LSTM was used to achieve an accuracy of 97.6% for 14 gestures and 91.4% for 28 gestures. Similarly, an accuracy of of 86.1% was achieved in [80], where a hidden Markov model was used to classify 24 gestures.

It was noted that the version of the controller used in the studies would have required significant updates to be a viable choice for sign recognition. In [78], the device was deemed to be unsuitable for complex signs that require significant face or body contact. Similar problems were reported in [80], where finger tracking was affected by the limited field of view of the device.

A sensor fusion-based system, where the Leap Motion Controller and Kinect were used together for classifying 10 gestures from the American Sign Language, was proposed in [82]. Both devices were used to locate a hand in their view separately, and different spatial features were extracted from the located hand. In accordance with data level sensor fusion, the extracted features were combined and used as an input for a classifier. The classifier used in this system was a multi-class SVM, for which an accuracy of 91.3% was achieved. When the features of only one of the devices were used as the input for a SVM, the achieved results were lower: 80.9% for Leap Motion Controller and 89.7% for Kinect. The authors also note the tracking problems with the device. [82]

Dynamic hand gestures were recognised using a Kinect v2, a hand pose estimation algorithm, and a combination of a 3DCNN and ConvLSTM neural networks. Data fusion between the RGB, depth and hand pose modalities were performed by a weighted sum between the three modalities, where the different modalities are combined into a single RBG image. The resulting image was used as an input for the fusion network. The proposed fusion method was justified by a smaller network size, but the fused image was only compared to each of the modalities alone. The authors report a 92.4 % recognition accuracy when using a dataset with 20 subjects performing 10 gestures repeated for 10 times [83].

2.5 Human robot interaction

HRI is a study of robotic systems that are used by or with humans. In addition to improving understanding, the goal of HRI is to shape the way humans and robots

interact with each other [84]. While keyboard and mouse-based interfaces can be suitable in some use cases, a large part of the work is focused on finding alternative approaches, that are more intuitive and easier to use, on which the interface can be based. In [84], HRI is divided in two general categories: remote and proximate interaction. This section focuses specifically on the latter category, where the humans and robots are located in the same space at the same time. A selection of applications from the field is presented here.

A gesture-based interface for controlling a mobile robot was presented in [19]. The interface was based on detection and tracking of colour-based segments corresponding to the face and shirt of a person. Similarly, the gestures of a person were detected using the segmentation mask to calculate the angles of the arm segments. Both static and dynamic gestures were detected, which were used to command the robot to, e.g., follow the person or stop moving. The authors reported a speed of 0.25 m/s for the robot to be preferable over a faster speed of 0.45 m/s. The higher speed decreased the perceived accuracy when trying to position the robot, while the navigation worked as intended [19].

When moving in places shared with people, the speed and the behaviour of the robot should be configured so that the robot does not cause discomfort to the people. In a study conducted in Japan, a robotic wheelchair was perceived by pedestrians to be most comfortable when moving slower (0.8 m/s) than the average walking speed (1.0 m/s - 1.2 m/s) and when taking a socially preferred route, i.e., moving on the left-hand side of a corridor as it is generally taken by local pedestrians of the country [85]. Many aspects affect the walking speed of pedestrians, such as the local culture, the time of the day, the season, and the environment type [86], which the robot should adapt to.

Person tracking and identification are core functionalities of robots that are designed to follow people as a part of their support tasks. In [87], a monocular camera-based tracking and identification system was presented, which uses OpenPose, a publicly available pose estimation algorithm, to first detect the people in the robot's view. The location of these people is transformed to the coordinate system of the robot, in which they are tracked using unscented Kalman filter. The transformation uses a predefined ground plane to help with estimating the heights of the people. After the tracking step, the initialised target is identified using methods called Convolutional Channel Features and online boosting. The online boosting-based approach is argued to be crucial for identifying the target in changing lighting conditions. Depending on the used camera, the resulting method can track people up to 10 or 20 m. In an evaluation session, the resulting localisation error was less than 0.5m within this range [87].

A similar setup for following people was described in [88], with an application in building construction and maintenance. The main contribution was focused on utilising building information models for robot navigation in the construction site, but a second application scenario included following a person within the site. Here, the person detection was performed using YOLO v3 architecture together with LiDAR-based depth estimates [88]. Similar approach for person identification was used as in [87]. Small computing platforms with integrated GPUs have become

popular in mobile robotics. In [88], NVIDIA Jetson Xavier AGX was used to perform computer vision related computation, whereas NVIDIA Jetson TX2 was used in [87].

A person-following controller for mobile robots with a simple hand gesture-based initialisation interface was presented in [21]. By detecting hands of a person in the camera view of the robot, the robot was able to initiate the follow command when the person closed either of their hands. The person detection was based on OpenPose, but no identification, apart from Kalman filter-based tracking, was included. Instead, the robot had to be reinitialised to continue the following task [21].

Pointing by a hand or a finger is a natural way for humans to indicate a target or a direction of interest when communicating with other humans. It has been argued that pointing would also be an important tool for HRI, especially as it does not require any external devices. In [20], Kinect depth camera and a third-party library for joint detection was used to construct a system to interpret pointed directions as destinations for a robot. The destination is calculated as an intersection of a half-line, defined by the position and the pointing direction of the hand, and the ground plane. When a destination command is given, the user is also required to lift their left arm as a confirmation. This reduces the number of false positives when interpreting a direction as a command. The variance of the reported results grow as the user moves further away from the robot. This is said to be due to smaller amount of valid pixel data available in the depth image. However, and more interestingly, the variation increases more on the direction parallel to the pointed direction projected to the ground plane. This could be due to the angle α between the pointed direction and the ground normal approaching an angle of 90°. The distance parallel to the ground plane is related to α by $x = y\tan(\alpha)$, where y is the height of the hand from the ground plane. The distance approaches infinity when α approaches 90°, as does the derivative $\frac{d}{d\alpha} y\tan(\alpha) = y\sec^2(\alpha)$. This means that the effect of small perturbations in α to the measured distance $\hat{x}$ increases with x. The study only varied the distance of the user to both the robot and the target point simultaneously, which makes it hard to separate the sources of error. Furthermore, a single target point was used throughout the study, giving only a partial understanding of the performance of the system. [20]

A different approach for human localisation was taken in [22]. The authors of the study used a wearable IMU sensor on the wrist to determine the pointing directions r_i^H in the local coordinate system of the user. The localisation is based on the user pointing at a moving robot, whose odometry is known in its own frame of reference. N of the robot positions P_i^R can then be collected with r_i^H to form a set of pairs C:

$$C = \{r_i^H, P_i^R\}_{i=1}^N$$

With an estimated transformation matrix T between the robot and the human coordinate systems, the robot locations P_i^R can be transformed into locations P_i^H in the human coordinate system as $P_i^H = TP_i^R$. Now half-lines q_i^H can be defined as originating from the corresponding positions of the hand and pointing towards the robot. The angles between q_i^H and r_i^H can then be calculated and an error function

can be defined as:

$$\theta\left(T, C\right) = \frac{1}{N}\sum_{i=1}^{N} \angle(r_i^H, q_i^H),$$

where $\angle(...)$ represents the unsigned angle between the directions of the half-lines. By minimising this error function for a given set C, an estimated transformation matrix can be obtained. Furthermore, a robot can be selected from a group of multiple robots, by choosing the one with the smallest error. This, however, requires that each of the robots is moving on a unique trajectory. When trying to identify the robot the user is pointing at from a group of two robots, the authors report that a duration of 5 s is sufficient for successful identification in most of the cases. The speed of the robot should influence the required time, but it was not discussed in the study. [22]

A subsequent study [89] then built upon this localisation approach, where it was used as a base for triggering the interaction and for giving commands to the robot. The interaction was initiated by pointing at the selected robot from the group of robots present in the scene. A robot for which the error function yields a value below a given threshold is then selected for interaction. Once the robot is selected, the system can use the estimated T to map destinations to the robot's coordinate system, towards which the robot can then move. The destinations are acquired in the same way as before, i.e., by calculating the intersection of the half-line and the ground plane.

The authors experimented with two types of closed-loop feedback. The first version used a sufficiently fast robot, whose position itself acted as the feedback to the pointed direction. The second version used a laser pointer attached to a turret to give the feedback of the position. To evaluate the effect of the feedback, the authors measured the pointing accuracy from a distance of around 2 m. Without any feedback, the error in the reconstructed pointing location was roughly 0.5 m on average. With the laser pointer enabled, the error reduced to below 0.1 m, but it took the users longer to reach this.

In another experiment, the subjects were told to fly a drone between predefined landmarks using the proposed and a joystick interface. The authors report comparable results between the interfaces, but using the joystick was still found to be faster. However, the subjects were used to the joystick interface but had no previous experience with the pointing interface. This speaks for the intuitiveness of the interface. [89]

One disadvantage of this approach is that the user is required to stay still after the interaction has been initiated. The system is not aware whether the user has moved after T has been calculated and it is not straightforward to extend it to constantly update T. On the other hand, the system does not require constant line of sight like camera-based systems do.

3 Methods

3.1 System description

To successfully implement the gesture-based HRI interface described in this work, a significant amount of working time had to be allocated for setting up the underlying system. This first subsection introduces the chosen components for the system: the robot and its payload, the sEMG armband, and the software and hardware required for computation and communication between the components. The rest of this section walks through the developed modules and the evaluation performed to validate the interface.

Spot (Boston Dynamics) is a quadruped robot aimed for automated sensing and inspection and it is able to traverse rough terrain and climb stairs. By default, Spot features 5 pairs of monochrome stereo cameras around its body, which are used for localisation, path planning, and obstacle avoidance. In addition, Spot features a customisable platform, allowing separate modules, called payloads, to be attached to the robot. These include a gripper arm, a more powerful computer for computationally intensive tasks, and additional sensing solutions, such as LiDARs and thermal cameras. [90]

To enable automated navigation, Spot uses a system, which includes mapping of the surroundings, localisation of the robot within the mapping, and an autonomous traverse system. The system differs from other common simultaneous localisation and mapping (SLAM) solutions by that the created mapping is a graph of waypoints, and the robot's position is determined relative to one of these waypoints at a time. [91]

EMG measurements are usually done using either separate electrodes, with systems like Delsys Trigno Research+, which integrates EMG electrodes with IMUs, or HD sEMG systems. These systems generally provide high resolution signals with a large sampling rate, that is, at least 1 kHz. These, however, can be quite inconvenient to use outside of laboratory conditions and require extensive preparation.

Another option is to use an sEMG armband that usually features eight evenly spaced electrodes around it. Only a few commercially available sEMG armbands are available at the time of writing. The first, and most commonly used in the scientific community, is the Myo armband (Thalmic Labs). Signals from the eight bipolar dry electrodes can be acquired at a sampling rate of 200 Hz with a resolution of 8 bits. In addition, the armband features an IMU [92].

A similar armband called gForcePro+, shown in Figure 3 is produced by OYMotion. The armband has an improved sampling rate of up to 1 kHz when the signal is sampled with an 8-bit resolution. The resolution can be increased to 12 bits, but this reduces the maximum sampling rate to 500 Hz [93]. It has been shown that reducing the sampling rate from 1 kHz to 500 Hz reduces gesture classification accuracy only a little [36], leaving both sampling modes as potential options for gesture classification. Similar to the Myo armband, gForcePro+ uses Bluetooth Low Energy (BLE) to enable communication with devices receiving the EMG data. Because of the better availability and improved features over Myo, and better portability over clinical devices, the gForcePro+ armband was chosen for this thesis.

Figure 3: The gForcePro+ sEMG armband used in this thesis. The numbers indicate how the eight sets of dry electrodes are mapped to the eight channels of the device.

An interface for gForcePro+ was developed for transmitting the measured EMG signals and to synchronize the signals with any output from Spot, such as images from its cameras. The interface was built on Python using ROS Noetic Ninjemys. The ROS interface allows the images and EMG signals to be easily recorded in a single rosbag file, which can later be replayed as if the signals were being generated at that time. The interface connects to gForcePro+ via BLE and publishes the received data to a ROS node.

ROS was used as a backbone for all the other software modules of the interface as well. The most important modules will be described in detail in the following sections, but several smaller modules were used throughout development, such as interfaces for selecting a point to focus on in the view of the robot and giving commands to the robot using the keyboard. These smaller modules were indispensable when testing and debugging other modules.

All the modules for the HRI interface were deployed to a NVIDIA Jetson Xavier NX Development Kit, running Ubuntu 20.04 included in JetPack 5.0.2. The Jetson Xavier was attached to the robot with a 3D-printed casing, on top of any already existing equipment. The Jetson Xavier was connected to a Spot CORE; another computer responsible for computation that does not require a GPU, via a network switch. The setup enabled wired communication between Jetson Xavier and Spot CORE, while allowing a laptop to be connected to either of the computers for debugging and real-time visualisations. This way, Spot was able to carry all the hardware required for running the interface. The attached devices are shown in Figure 4.

3.2 Operator detection

To interact with the robot using the proposed interface, a person must wear the sEMG armband so that any performed gestures can be detected. Using this constraint, we can identify the correct person, i.e., the operator, from the view of the robot. Using

Figure 4: The devices attached to Spot. Left: From top to bottom: NVIDIA Jetson Xavier NX Development Kit in a 3D-printed casing, Ouster OS0-128 LiDAR, Velodyne Puck LiDAR, and Spot CORE. The network switch is in the bottom right of the image below the Spot Arm. While the Velodyne Puck was used by Spot for sensing the environment and the Ouster OS0-128 could be used for performing additional scans of the environment, the two LiDARs are not directly related to this work. Right: Flir thermal camera attached to the arm.

neural networks, we can detect both people and gForcePro+ armbands in an image and try to use their spatial relationships to identify the correct operator.

In order to identify the operator from the other people possibly in the view at the same time, we can assume that at some moderately short time interval the armband should be visible and thus, any person who has not been seen to wear an armband is not a candidate to be the operator. The reverse is not as straightforward. It is possible that there are multiple items present in the scene that are either gForcePro+ armbands or resemble the armband enough to confuse a neural network. That is why it does not necessarily mean that if a person is detected to be wearing an armband, that they actually are the operator.

The neural network used for detecting people and gForcePro+ armbands is based on the s-variant of YOLOv5 [23]. The architecture of YOLOv5 is shown in Figure 5.

The training data consists of a mixture of images, where the first part is a subset of the COCO dataset [95], where at least one person is labelled, whose bounding box was sufficiently large. The second part is a collection of images featuring a person wearing the gForcePro+ in various poses and environments as shown in Figure 6. The dataset was heavily unbalanced; there were 47997 images extracted from the COCO dataset containing only people, whereas there were only 379 images containing the armband in limited number of environments and people wearing it. Most of the pictures containing the armband were taken for this dataset while some were found using image search from Google and Baidu. To address the issue of unbalanced datasets, the images were sampled using the inverse of their mean average

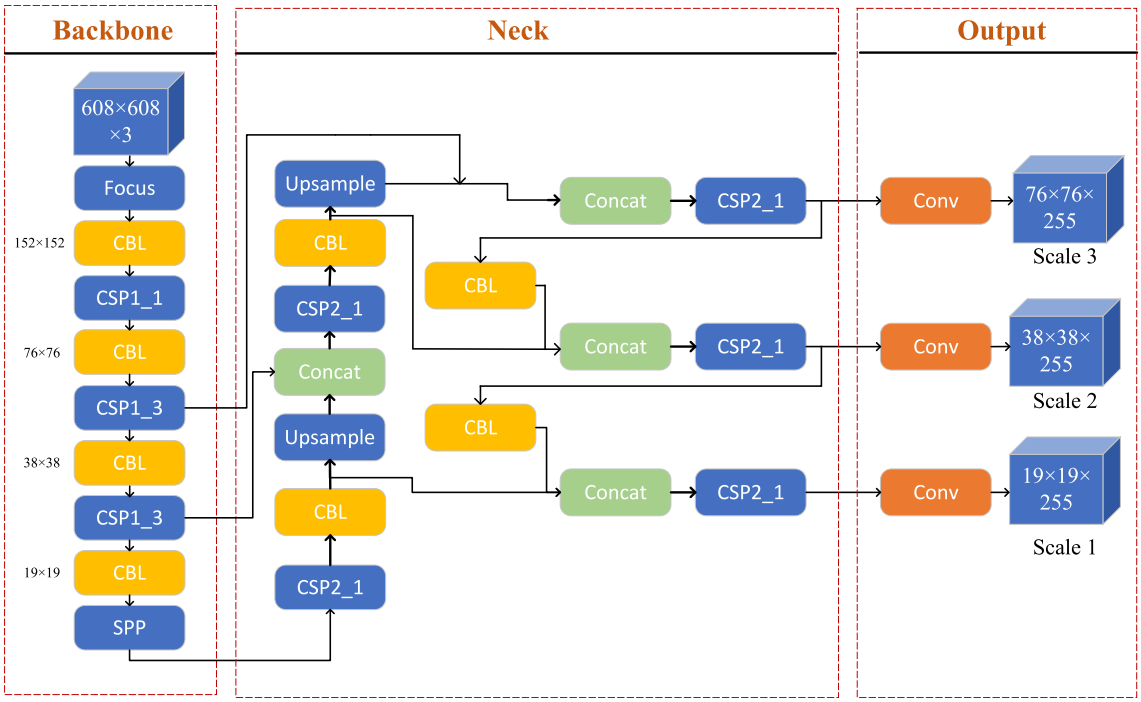

Figure 5: The architecture of YOLOv5. Adopted from [94]

precision (mAP) score, resulting in images with low-mAP objects being sampled more frequently. The images were augmented using the inbuilt augmentation method of YOLOv5, which builds a mosaic out of a original image and 3 randomly sampled images and apply lightness and saturation changes to the final image along with spatial transformations [23].

The neural network was trained using transfer learning with the pretrained weights of YOLOv5s. The first 10 layers, i.e., the backbone seen in Figure 5, were frozen so that retraining would not affect the weights of these layers. They are responsible for low-level feature extraction and are assumed to be optimally trained. The new model was trained using the Adam optimizer for 30 epochs with a batch size of 16.

In this work, the operator is identified using the detected people and the detected armbands, and calculating a score for each person using the Equations 5.

$$\begin{aligned} d_{ij} &= \|\mathbf{p}_i - \mathbf{g}_j\|_2 \\ s_{ij} &= 1 - \min\left\{0,\ \max\left\{\frac{d_{ij}}{p_{ix}},\ 1\right\}\right\} \\ w_i &= \sum_{j=1}^{n} s_{ij} q_j, \end{aligned} \tag{5}$$

where d_{ij} is the Euclidian distance between vectors $\mathbf{p}_i$ and $\mathbf{g}_j \in \mathbb{R}^2$, the centers of the bounding boxes of a detected person i and a detected gForcePro+ armband j, s_{ij} is an intermediate score calculated for a pair i and j. w_i is the weighted score for person i, and q_j is the confidence score of armband j estimated by a neural network.

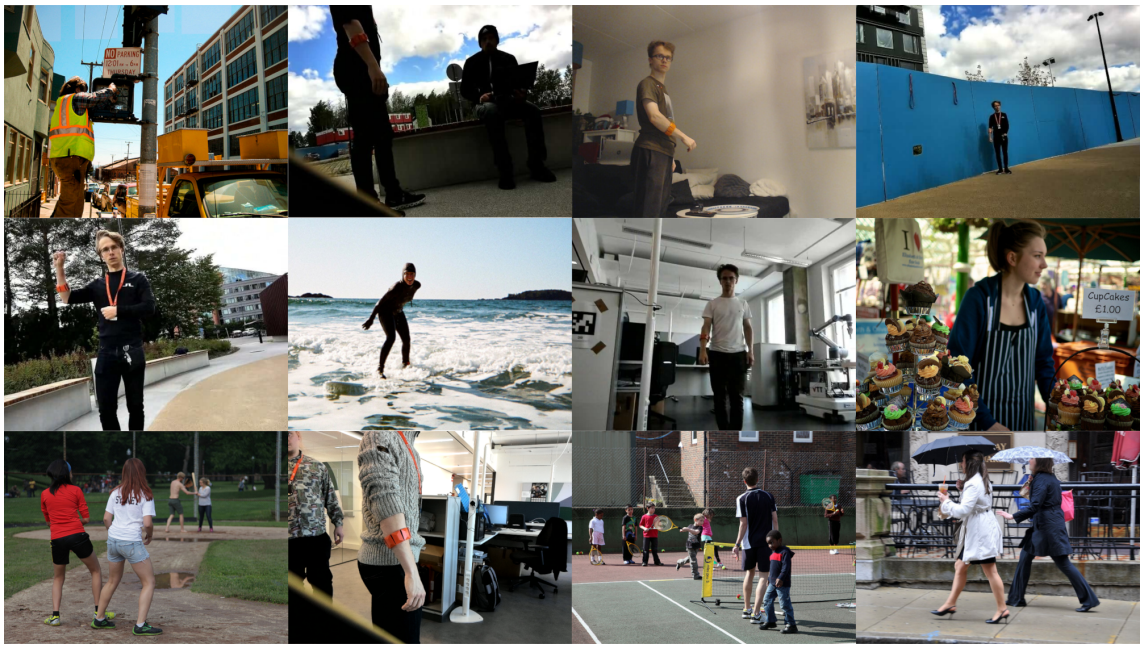

Figure 6: A sample of the dataset used for training the object detection neural network. In addition to a subset of the COCO dataset [95], which contains only images where people are present, a set of pictures where people are wearing the sEMG armband was collected.

p_{ix} is the x-component, i.e., the first element of vector $\mathbf{p}_i$, which is used to limit the maximum distance a detected armband can be from a detected person to have an impact to the person's score. q_j is used to reduce the impact of uncertain armband detections.

The detected person i with the highest score w_i is classified as the operator. The score is updated each frame, where at least one armband is visible. However, to be able to update the score of each detected person, detected people of a previous frame need to be associated to the new instances of detected people in the new frame.

The data association is solved using global nearest neighbour method, which associates detections by minimising the global cost. The cost function used is Euclidian distance. Individual tracks are maintained by a Kalman Filter-based tracker with a (nearly) constant velocity state-space model, where the state $\mathbf{x}$, the state transition model $\mathbf{F}$, and the observation model $\mathbf{H}$ are:

$$\mathbf{x} = \begin{bmatrix} p_x \\ v_x \\ p_y \\ v_y \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

where p_i and v_i are the position and velocity of a detection along the i-axis, and Δt is the time passed between two consecutive frames. The detected people are associated to already existing tracks and if no matching track is found a new track is created.

There are some restrictions when it comes to the lifespan of a track. A new

track cannot be initialised within a searching radius r_s to limit the impact caused by duplicate detections that sometimes occur when using neural networks to create the initial detections. A track can be lost if no detections are associated when processing a frame, and if too many frames pass without an associated detection, the track is deleted. To minimise the recovery time when a track representing the operator is lost, the number of frames before a track is deleted should be set to a small value; here a value of 3 frames is used.

The score calculated by equation 5 is attached to the corresponding track and the score is updated by linear interpolation each time a new frame with at least one armband visible is received. In addition to requiring the chosen track to have the highest score of all tracks active at the given time, a constant threshold of 0.2 was also used to make the system more tolerant to false associations.

The images used for operator detection are captured using the cameras in the Spot Arm payload, which also features a depth camera. The visual images are compressed using Theora video compression and the depth images are compressed to JPEG images to improve the frame rate. The depth images are projected to match the visual frame.

When the operator is detected in a received visual frame, the matching depth frame can be used to estimate the distance to the operator. The bounding box of the operator in the visual frame can be applied to the projected depth frame, from which the content within the bounding box can be extracted. The final estimate for the distance is calculated as the median of the depth information withing the bounding box, as shown in Figure 7.

To get the complete estimation of the operator position in the camera coordinates, the location of the operator is transformed using the pinhole camera model. The centre of the bounding box is selected as the location in pixel coordinates. The transformation is done by backprojection using the intrinsics matrix of the camera. First, the 3D point describing the operator location in camera coordinates is transformed to the hand coordinate system of the Spot Arm, then to the coordinate system of the Spot body and finally to the world coordinate system, which is maintained by robot's navigation system. The position in world coordinates is used to define different commands for Spot to execute.

The depth camera located in the arm of the robot is the primary modality used for estimating the distance to the operator. The quality of the estimate, however, depends on the distance and at some point, as the distance increases, the estimates are no longer received. To better understand this relationship, the robot was teleoperated in an office environment to first walk away from a person in reverse and then walk forwards toward the person. A dataset containing the depth camera data was collected in this process, where the ground truth distance was obtained from the odometry of the robot. The initial offset was obtained using the depth camera, which faced the person for the entire time. The dataset was used for calibrating a distance estimator based on the height of the detected bounding box.

The capabilities of the operator detection system were evaluated at three locations as a function of distance. The robot was teleoperated in a similar manner to walk back and forth away from a person, while keeping the person in the view. The

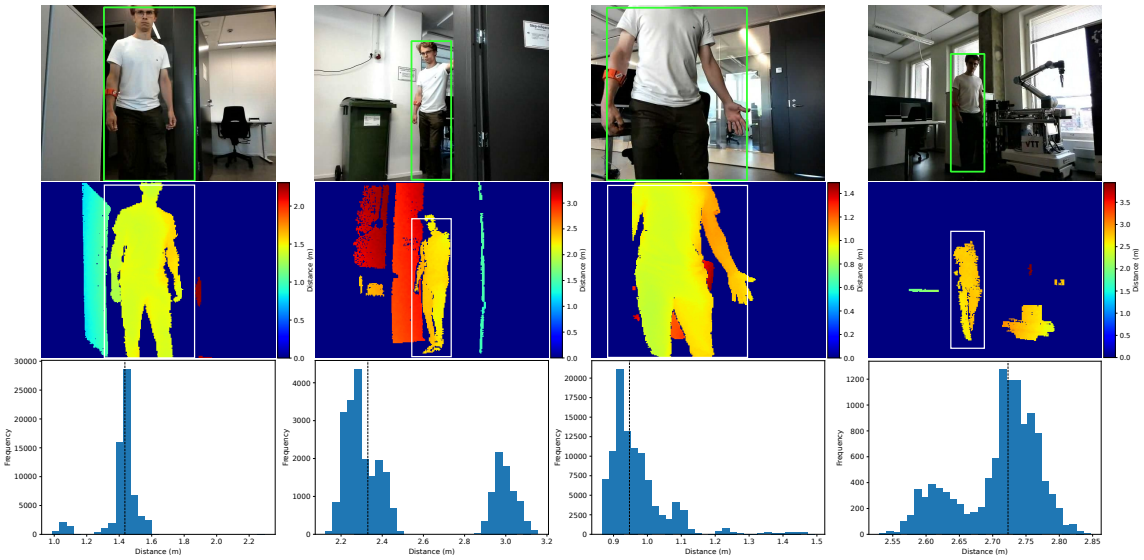

Figure 7: Top row: Visual frames captured using the RGB camera in the Spot Arm, where the bounding boxes of a detected person are overlayed. Middle row: False-colour images of depth frames projected to match the visual frames. Bottom row: Histograms of the depth information within the bounding box. The median, which was used as the distance estimate, is marked with a vertical line. Pixels with value NaN are mapped to zero in this visualisation and ignored for the depth estimate.

locations of these evaluations are shown in Figure 8. From these measurements, the relationship between the performance of the depth cameras for operator distance estimation was evaluated in these different environments, as well as the range of the BLE connection between the gForcePro+ armband and the Jetson Xavier attached to the robot.

Figure 8: Distance related measurements with Spot at different locations. Left: Indoors location at an office. Middle: Outdoors location in direct sunshine. Right: Outdoors location in the shade.

3.3 Gesture detection

3.3.1 Parameters for the sEMG armband

The gForcePro+ sEMG armband supports different sampling rates up to 1 kHz, and two ADC resolutions: 8 and 12 bits. When using 12-bit resolution, however, the 1 kHz rate is not supported, and a lower rate of 500 Hz is used instead [93]. This limit is caused by using a single BLE chip for wireless communication of all eight sEMG channels. The sampling rate is higher than that of Thalmic Labs Myo armband, which was limited to 200 Hz with 8 bits ADC resolution, but lower than wireless medical grade sensors, such as Delsys Trigno [96].

Delsys Trigno is a wireless sEMG device that records one channel per device. Multiple of these devices can be used together to record multiple channels simultaneously. It can provide a sampling rate up to 4.3 kHz with a 16-bit resolution even though it uses the same BLE 4.2 standard that is used in gForcePro+ [96]. This is likely made possible by transmitting only one channel per BLE connection as opposed to the eight channels of gForcePro+. At 4.3 kHz, Trigno would be transmitting 68.8 kb/s, which is close to the rate of 64 kb/s transmitted by gForcePro+ when sampling with the 1 kHz, 8-bit mode. The calculated bit rate of gForcePro+ is decreased to 48 kb/s when using the mode with 500 Hz rate and 12-bit resolution. Empirical tests allowed 650 Hz rate to be used with 12-bit resolution, which corresponds to 62.4 kb/s, but to follow the guidelines of the manufacturer, this mode was not used in this thesis. The Myo armband transmits at a bit rate of 12.8 kbit/s.

To evaluate the gesture detection potential of the two modes of gForecPro+ (8 bit at 1 kHz and 12 bit at 500 Hz), the data from Ninapro DB2 was used to approximate these operating modes. DB2 consists of 40 subjects performing 49 gestures repeated 6 times each. The sEMG signals are recorded using Delsys Trigno system with 2 kHz sampling rate and 16-bit resolution [50]. All combinations of 6-, 8-, 12- and 16-bit resolutions and 200, 500, 1000 and 2000 Hz sampling rates were also evaluated for a more complete picture. These include 8 bits at 200 Hz, which are the operating parameters of the Thalmic Labs' Myo armband [97].

The sEMG signals of DB2 were subsampled at the chosen sampling rates, but first a low-pass filter was applied with a cut-off frequency of the corresponding Nyquist rate. The resulting signals with different sampling frequencies were further processed by clamping the signal values to a range of ± 1.04 mV, which is the input range of gForcePro+. The effect of clamping was evaluated by comparing the acquired signals to the unclamped signals. Finally, the samples were converted to 8-, 12- or 16-bit integers, to simulate the possible ADC resolutions.

The signals were cut into segments containing a single gesture each. As a form of data augmentation, the signals were randomly trimmed to segments with a length of 11 s. A new trimming of the segment was produced for each epoch of training, whereas testing was performed with the original segment. As the original segments were chosen to have considerable overlap, this method is closely related to the SW method. SW has been shown to be one of the most beneficial data augmentation methods that has been applied to sEMG signals [68] and other biosignals such as EEG [64].

An MLP was used to classify the gestures from all the subjects simultaneously. The segments were divided into training and test sets, where one of the six repetitions were assigned for testing and the rest for training. This was repeated for each of the six repetitions, thus forming a leave-one-out cross-validation (LOOCV) scheme based on the repetitions. The results are reported as the average of the six repetitions.

The MLP takes feature vectors as input and outputs probabilities for different gesture classes. The hand-crafted input features include MAV, waveform length (WL), Willison amplitude (WAMP), ZC, mean absolute value slope (MAVS) and mean frequency (MNF), which were recommended in [5]. The input signals were split into windows, from which the features were calculated. The window length was chosen to be 200 ms for all sampling frequencies, to keep the response time of the system within acceptable limits for real-time applications [42] and to better ensure comparability between the simulated operating modes. Thus, the resulting window lengths were 40, 100, 200 and 400 samples for sampling rates of 200, 500, 1000 and 2000 Hz, respectively.

Before training the models, the mean and standard deviation of the features were calculated over the training data for each parameter pair, so that the features could be normalized to have a mean of 0 and a standard deviation of 1. Furthermore, the weights of the models were initialized using the PyTorch implementation of Kaiming initialization [98].

After calculating the six normalized features from a given window for all 12 channels, the 72 features were concatenated as a single feature vector and fed to the MLP. The MLP consisted of 3 hidden layers, sized 2048, 1024 and 512, with ReLU as the activation function. Dropout layers with a dropout probability of 0.2 was used after each activation to reduce overfitting to the data. All the MLPs with different sampling rates and resolutions were trained for 50 epochs with a batch size of 512 using Adam optimizer. The initial learning rate was set to 5.5×10^{-4} with an exponential learning decay with $\gamma = 0.93$.

The previously described model takes the features of a single window as the input and outputs gesture class probabilities. As the window is 200 ms long, only a fraction of a complete gesture is analysed per iteration, which can lead to noisy predictions. To get the final prediction of a complete gesture, the most probable class was picked from each window and the number of occurrences were counted for each class. The most common class was then picked as the final prediction, where the null class, i.e., no gesture, was ignored as there are no trials with no performed gesture included in the dataset.

3.3.2 Neural networks

Using the operator detection system described in Section 3.2, the robot is able to identify and follow the operator. This alone is not sufficient, however, as the operator has no way to communicate with the robot, apart from altering their relative position. There are various possibilities for providing the means of communication, such as a hand-held controller, voice commands, and gestures, the last of which can be detected through visual means or wearables. In this work, a wearable sEMG armband called

gForcePro+ was used to enable the required communication.

Multiple neural network architectures were trained and evaluated using a dataset described in section 3.5.1. The models include an MLP, CNN, MLP-LSTM and CNN-LSTM models, whose architecture diagrams are shown in Figure 9. For the MLP and MLP-LSTM models, a feature set was used to extract features of the sEMG and IMU signals, whereas raw signals were used for the CNN and CNN-LSTM models.

The MLP was similar to the one described in Section 3.3.1, including the applied feature set, which contains features MAV, WL, WAMP, ZC, MAVS and MNF. However, this model included 12 additional features; MAV and VAR, which were calculated from each of the x, y and z components of the acceleration and gyroscope signals received from the IMU. The window size was kept at the same value of 200 samples as the 1000 Hz models of the Ninapro data. The loss function was also changed from cross entropy loss to MSE loss, as the task was changed from classification to regression.

The CNN model consisted of two convolutional modules and three fully connected layers. The first convolutional module consisted of 1D convolutional, max pooling, and ReLU layers, and it was responsible for extracting features from the EMG signals. Each of the eight channels were processed separately by the same module and the output was concatenated afterwards. This way the model needs to learn only one set of weights for processing a generic EMG channel. Both the current and the previous EMG windows were given as the input. The IMU signals were processed by a similar, albeit smaller, convolutional module, where all six channels were used as an input in the same pass. The details of the CNN and the other models are shown in Figure 9.

After training, both MLP and CNN models were frozen and used as backbones for two LSTM hybrid models: LSTM-MLP and LSTM-CNN. The last fully connected layers from the baseline models were removed to provide a richer presentation of a given state of the hand for the LSTM. Both models used one unidirectional LSTM layer with a hidden state size of 32 and a fully connected layer, which formed the prediction.

The four models were trained separately for each subject, using the first three sessions for training and the fifth for validation. The best performing model was selected based on the regression MSE on the validation set.

The effect of donning and doffing was investigated by replacing the recordings from the third session of the training set with the recordings from the fourth session and retraining the selected models with these subsets of the data. The fourth and fifth sessions were recorded without taking the armband off in between, and are thus considered to be effectively from the same session.

Reusing the convolutional layers for processing the eight channels of the EMG signal separately was validated as follows. The input size of the convolutional module responsible for extracting EMG features was increased from two layers and 200 samples to 16 layers and 200 samples. Now, the signals could be passed through the module at the same time in the same way as the six IMU signals. While the concatenated layer previously featured 6336 elements, it was now reduced to 960 elements, which reduces the overall size of the model considerably.

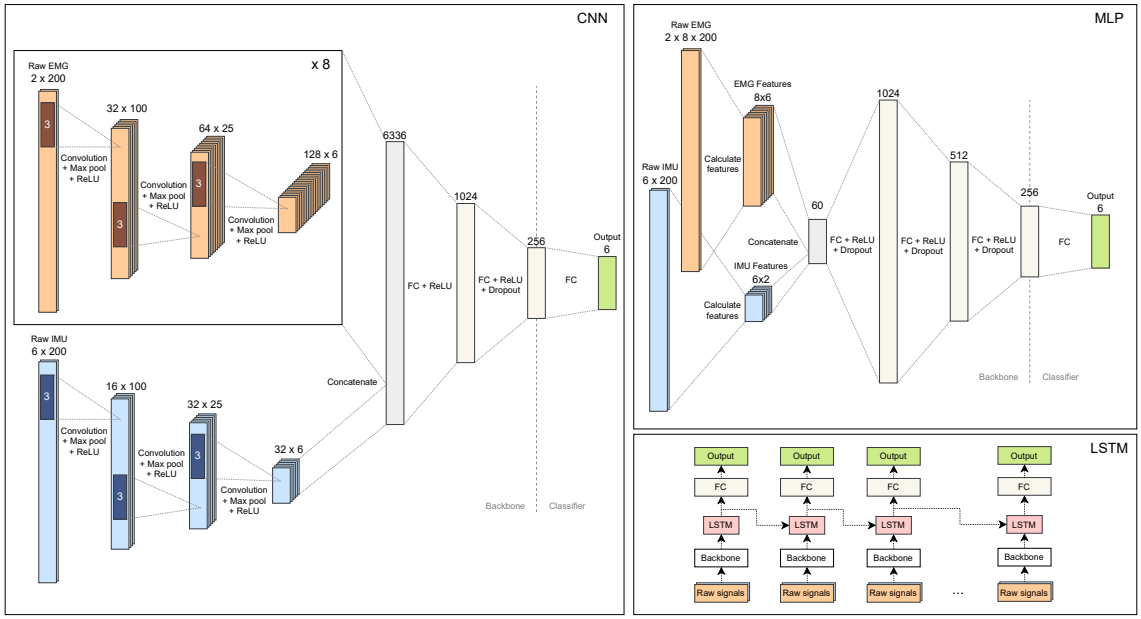

Figure 9: The architectures of the four models, where the two LSTM models were implemented using MLP and CNN as backbones. The classifier layer included in the MLP and CNN models were removed when the models were used as backbones for LSTM-MLP and LSTM-CNN.

All the models were trained using the Adam optimiser with a batch size of 64. A learning rate of 5.5e-4 was used for the MLP and CNN models, whereas the LSTM models were trained with a learning rate of 5e-2. The MLP and CNN models were trained for 50 epochs and the LSTMs for 30 epochs, as the backbones for these models were already trained and frozen. Exponential decay of the learning rate was applied with $\gamma = 0.965$ when training all the models. The models were trained using a system with NVIDIA Titan RTX and Intel i9-9900K.

3.4 Finite-state machine

In other works, where sEMG-based HRI interfaces have been implemented to control mobile robots, the detected gestures were mapped directly to the move commands of the robot [15, 16]. This is not ideal in this case, as the reliability of sEMG-based gesture classification is affected by muscle fatigue [18]. By requiring the user to provide continuous input with their hand should result in faster muscle fatigue. Instead, the gestures of this work are mainly used as triggers that alter the current behavioural state of the robot.

The behaviour of the robot was determined by a finite-state machine depicted in Figure 10. Keeping the state machine as simple as possible was important, as the operator needs to be able to interact with the system without being able to verify the currently active state. Changing the movement state from sitting to standing to the different available walking speeds were performed using wrist extension, while wrist flexion moves the state to the opposite direction.

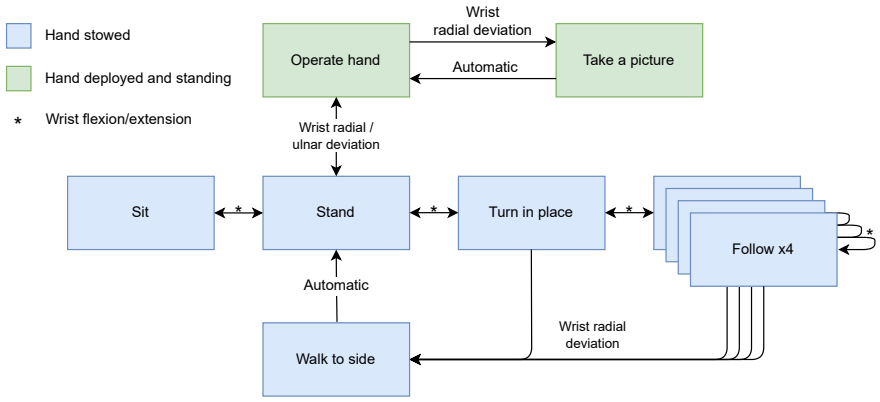

Figure 10: The state machine of the robot. For visualisation purposes, the transitions corresponding to fist and hand open gestures are not displayed. The fist gesture halts the movement of the robot from any other state, whereas the hand open gesture jumps to the second speed of the follow state. A transition between the states is triggered, when a cumulative value of the corresponding proportional gesture reaches a predetermined threshold.

Radial deviation gesture was used to change from standing to operating the arm. In this mode, the robot stays still, while keeping its hand pointed towards the operator. By performing another radial deviation gesture, the robot held the hand still for 3 s before taking a picture with an RGB and a thermal camera, which was also attached to the arm. Returning to the standing state was performed using ulnar deviation of the wrist.

While the entire state machine could be traversed using the four wrist gestures, traversing the state machine one state at a time could prove to be tiring for the operator. To alleviate this, the two remaining gestures were used for jumping directly to certain states. The hand open gesture initiated following in the second speed of the Follow state, whereas the fist gesture was used to command the robot to halt movement immediately from any state, or in other words, transition to the Stand state. If the BLE connection to the armband was lost for any reason, including the user turning the device off, a similar halt signal was sent as a precaution.

3.5 Evaluation

3.5.1 Gesture dataset

There are multiple publicly available datasets for sEMG-based gesture detection, such as the Ninapro [50] and putEMG [67] datasets, but, to the best of the author's knowledge, none for this particular device. A small dataset was collected to enable training a gesture detection neural network.

Based on the literature [4, 50, 99], six gestures were chosen for the interface and are thus included in this dataset. As in [4], the gestures include hand in a fist and hand open (or finger abduction), wrist extension and flexion, and wrist radial and ulnar deviation, in addition to a resting pose. The gestures are shown in Figure 11

To better utilise the IMU built into the armband, two gestures; hand in a fist and hand open, were modified to include lifting the hand to a vertical pose at the height of the subject's head. The rest of the gestures were performed with the hand held pointing downwards on their side. In addition, idle sequences with no gestures were recorded, where the subjects were instructed to pace back and forth within a radius of a few meters.

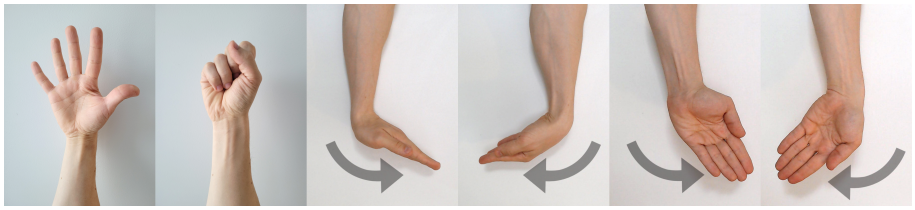

Figure 11: Gestures included in the dataset: hand open, fist, wrist extension, flexion, radial deviation and ulnar deviation.

Real-time feedback was provided for the four wrist-based gestures, whereas fist and hand open gestures were performed with only visual cues. The angle of the wrist was calculated from a web camera view using MediaPipe Hands [100] for hand detection. The normalised angle was displayed for the subjects as shown in Figure 12 with the rest of the view that was presented to the subjects during recording. OpenCV, PyQtGraph and ROS were used in addition to MediaPipe to implement the recording software.

The dataset was collected in the following manner. The duration for each recorded gesture was sampled between 2 s and 4 s, and an idle padding was applied so that each repetition lasted 5 s, with a 0.5 s additional pause between repetitions. Each of the six gestures was repeated for eight times per session and a total of five sessions were recorded with each subject. The last session differed from the others in that it was recorded directly after the fourth session, without taking the armband off. This way, the effect of donning and doffing could be quantified better.

Four male subjects participated in the sessions. Three of the subjects were right-handed and one was left-handed. The sEMG armband was placed by the subjects themselves on the right hand either at the 2/3 point from the wrist towards the elbow, or at the maximum distance away from the wrist if the circumference of the armband was too small for the chosen point to be used. The zeroth channel of the device, shown in Figure 3, was placed at the centre, on the ventral side of the forearm. The subjects were instructed to place the armband approximately at the same location every time. Slight variation was intended, however, as it would also be expected when the armband is used outside of laboratory conditions. On average, the centre of the armband was placed 15.1 cm away from the radiocarpal joint and the standard deviation in the location within subjects was 0.86 cm. The locations were estimated from images taken before the sessions, sample of which is shown in Figure 13. The sessions were divided on multiple days for each subject. Two sessions were allowed to be performed on a same day with at least 45 minutes separating them, except for the last session, which was recorded immediately after the previous

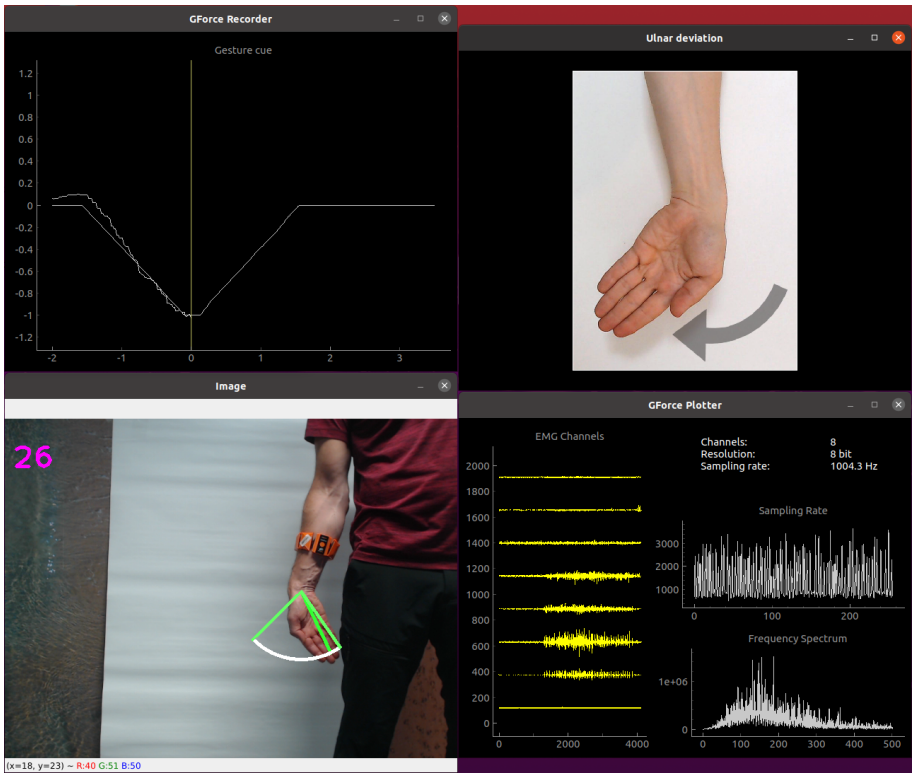

Figure 12: The view presented to the subjects while recording the gestures. Top-left: Gesture cue (straight line) and feedback (squiggly line), top-right: image of the gesture being performed, bottom-left: camera view, from which the feedback is calculated, and bottom-right: channel contents, sampling rate and frequency spectrum of the recorded signal.

one. On average, there was 29.6 h between each session. The subjects were allowed to get familiar with the system before performing the gesture sequences.

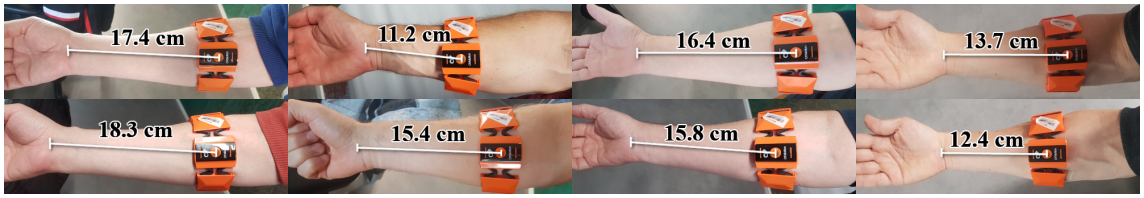

Figure 13: The sEMG armband was placed by the subjects. Some variation in the location was present, whereas the rotation of the device around the forearm was quite consistent. The pictures of the subjects are presented in no particular order.

3.5.2 Complete system evaluation

A proof of concept for the proposed interface was carried out by one expert user. The interface was used to instruct the robot to follow a operator along a path and to perform a set of actions. The planned sequence included taking control of the robot,

walking, stopping the robot, taking a picture from a point of interest, commanding the robot to turn and stop next to the operator, and to lie down. Completing these actions using the interface required performing a minimum of 12 gestures, listed in Table 1 with the corresponding actions. The sequence was repeated for five times.

Gesture detection was performed using the MLP-LSTM network trained previously using the described dataset. The proof of concept was conducted two days after recording the last gestures of the dataset.

Table 1: Evaluation sequence.

Task description	Gestures
Stand up and follow	1x Hand open
Increase speed to maximum	2x Wrist extension
Stop	1x Fist
Extend arm, keep the operator in view	1x Radial deviation
Take a picture	1x Radial deviation
Stow arm	1x Fist
Follow	1x Hand open
Increase speed to maximum	2x Wrist extension
Walk to the right-hand side of the operator, turn around and stop	1x Radial deviation
Lie down	1x Wrist flexion

The sequences were performed outdoors on a 110 m long footpath open to other pedestrians to evaluate the interface in a realistic environment. Additional gestures were performed when needed, e.g., stopping the robot when other people were passing by. The environment around the path is shown in Figure 14.

As the sequences contained some unplanned events, all the performed gestures were included in these statistics, while the intention of the gestures, i.e., the ground truth, was determined from a video recorded during the session. The execution of a gesture is considered to be successful, if the gesture detected by the interface matches the intended gesture and the robot performs the corresponding action.

The gesture detection accuracies and the F-scores are reported for each gesture type separately and for all gesture types combined. In addition, total number of performed gestures and the average trigger times for each of the gesture types are reported, as well as the average duration and walking speed of the complete sequences.

Figure 14: Views from the path the evaluation sequences were performed on. Left: Commanding the robot to follow, Middle: robot following the operator, Right: showing the robot where to take an IR image. A nitrogen container was chosen as the target of the IR imaging task, as the temperature differences were more prominent there than in the surroundings.

4 Results

4.1 Parameters for the sEMG armband

The MLP trained with Ninapro DB2 data achieved good results comparable to other, even state of the art, results. Figure 15 shows the training loss, validation loss, and validation accuracy for a single window and the validation accuracy for complete gestures. The testing was performed using LOOCV with each of the six repetitions of the gestures. Thus, the results are reported as the average of the six training iterations.

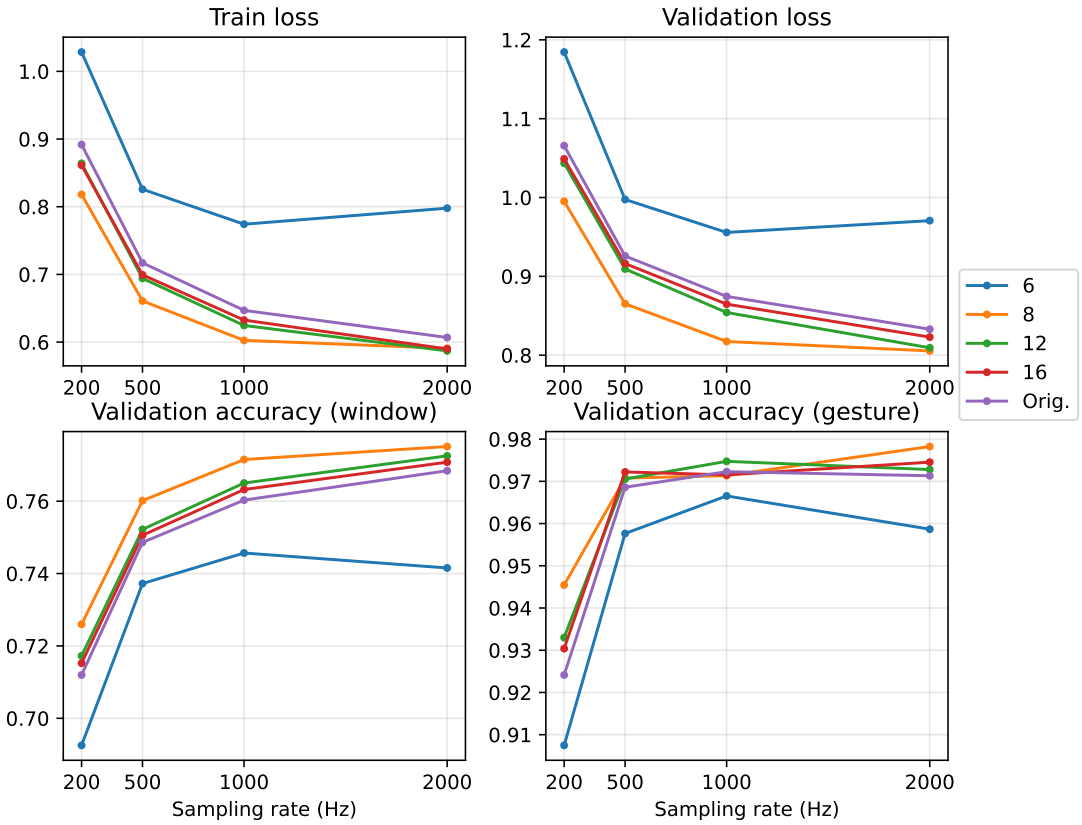

Figure 15: Training results of twenty combinations of sampling rate and ADC resolution. Individual lines indicate the simulated ADC resolutions tested in this analysis. The signal values with resolutions 6, 8, 12 and 16 have been truncated between ±1.04 mV, whereas "Original" uses non-truncated signals with 16 bit resolution.

The gesture detection accuracy is known to increase with sampling rate up to at least 1 kHz [6,36] and the results obtained here agree with the previous findings. The relationship between ADC resolution of the signal and the performance of the model is more complicated. While increasing the resolution from 6 bits to 8 bits improves the results, the results start to decrease when the resolution is increased further.

Another set of networks was trained to investigate the relationship between the truncation limit and the training results, while keeping the sampling rate constant

at 1 kHz. The gForcePro+ armband has an input range of ±1.04 mV, which was used as a baseline, around which the limit was varied. As shown in Figure 16, the best results are obtained by setting the input range between $5 \cdot 10^{-5}$ V and $1 \cdot 10^{-3}$ V. As before, the results are reported as the average of 6 iterations of LOOCV.

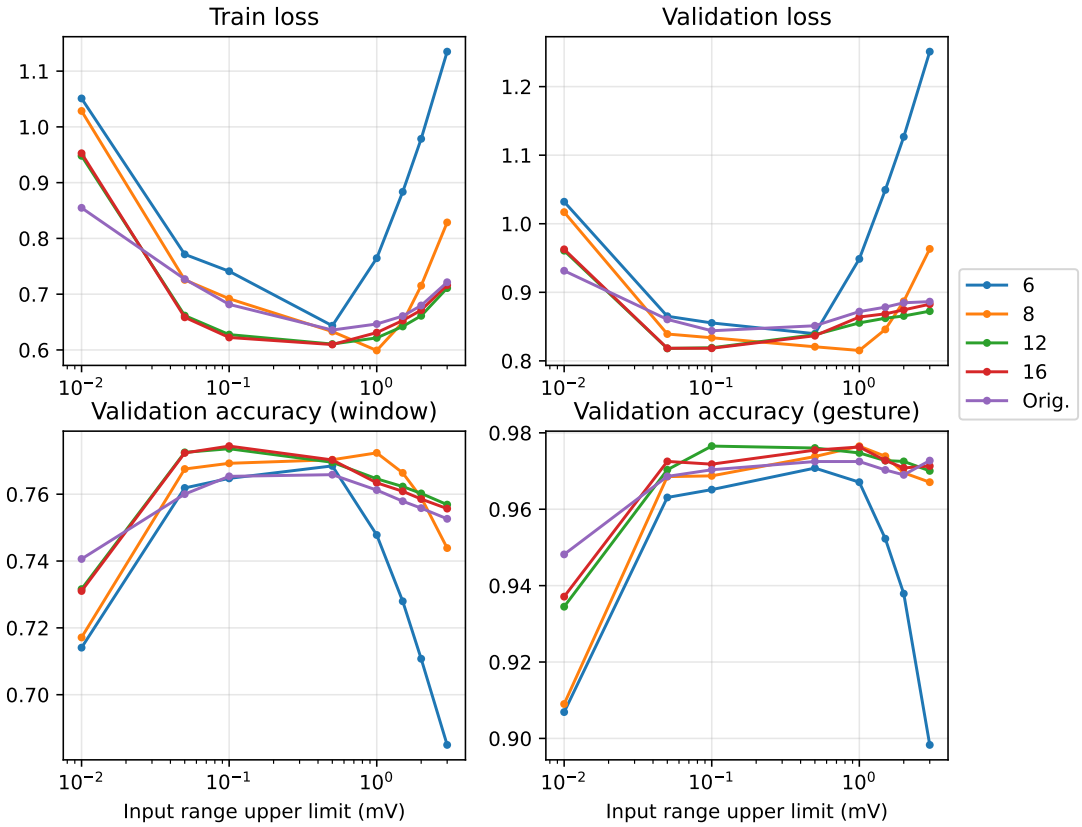

Figure 16: Comparison between different levels of signal truncation, ADC resolution and training results. The input ranges of ±0.01, ±0.05, ±0.1, ±0.5, ±1.0, ±1.5, ±2.0 and ±3.0 mV were evaluated. The results are plotted against the upper limit of the input range.

4.2 Interface calibration and evaluation

The results of the calibration for estimating the operator distance from the bounding box is shown in Figure 17. The number of valid samples received from the depth camera, i.e., the pixels of the image where a value other than NaN is returned, drops rapidly as a function of distance. The number is reduced to roughly 1% of the observed maximum by a distance of 5 m. As described in section 3.2, the depth image is aligned with the RGB image and only the samples within the bounding box of a detected person are included. The relationship between valid pixels and the distance is largely affected by the area of the bounding box, and for this reason also the ratio between valid pixels to all pixels is shown.

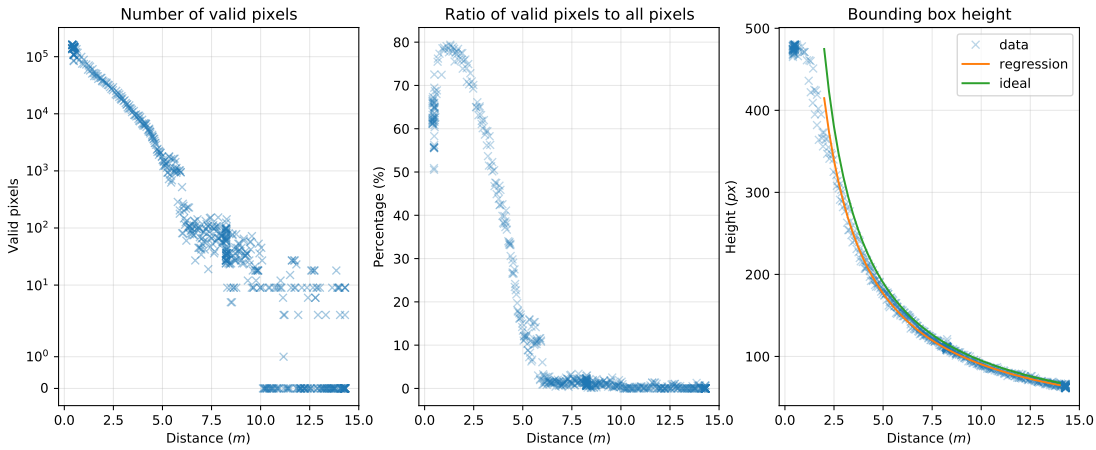

Figure 17: Depth camera performance with distance between the robot and a person. Left: The relationship between distance and the quality of the available distance samples within a bounding box of a detected person. Middle: Ratio of pixels for which a depth estimate is received vs all pixels within the bounding box. Right: Bounding box height vs. distance. A regression curve was fitted to estimate the distance from the bounding box height, whereas as the curve labelled as "ideal" shows the result obtained with a pinpoint camera model.

From this data, a boundary condition for accepting the distance estimate using the depth camera was set to $1 \cdot 10^3$ valid pixels, corresponding roughly to 5 m in the close-to-ideal conditions. An alternative distance estimator was implemented using the height of the bounding box of a detected person, which is inversely proportional to the distance. The estimator was calibrated using this dataset and the resulting regression is close to theoretical results as shown in Figure 17. In Figure 18, it is shown that the depth camera-based distance estimate . When the distance between the person and the camera is short, the feet or the head of the person, or both, are left outside of the view of the camera. This limits the bounding box-based distance estimator to be used only at distances at which the whole person is visible.

The performance of the depth camera as well as the sEMG data transmission over BLE was evaluated against distance between a person and the robot in three locations, shown earlier in Figure 8. In the two outdoor environments, the number of valid pixels dropped earlier, resulting in discardment of the results at shorter distances. The bounding box height-based approach was implemented for this reason, and to extend the detection range further.

The sampling rates were calculated as inverse of the time between samples, dt. Thus, the mean, and the 5th and 95th percentiles were weighted by dt, to prevent higher frequencies from dominating lower frequencies. The results are shown in Figure 19.

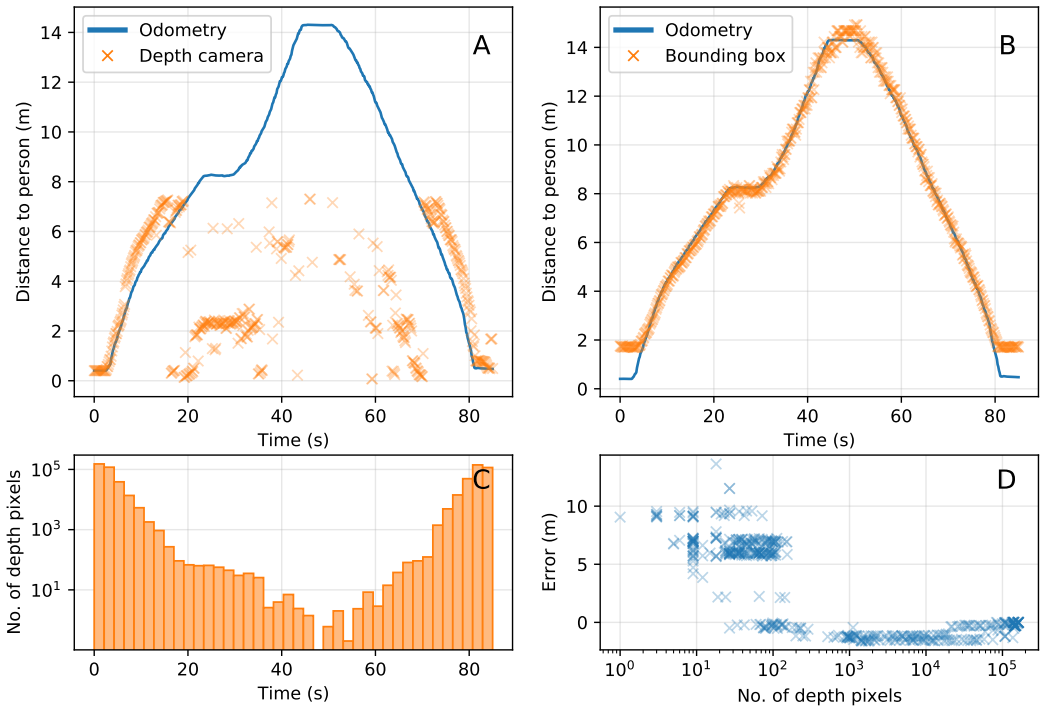

Figure 18: A: Robot odometry vs. depth camera-based estimates of the distance between a person and the robot. B: Robot odometry vs. bounding box height-based estimates of the distance between a person and the robot. C: The average number of depth pixels within a bounding box of a detected person corresponding to the data shown in A. D: The error in depth camera based distance estimates as a function of the number of depth pixels within the bounding box.

4.3 Gesture detection

The neural networks described in Section 3.3.2 were trained and evaluated using the collected dataset. The different models were trained using each of the subjects' data separately and combined together, resulting in four individual models and a multi-user model. The results for the multi-user model, that is, training time, number of parameters in the models, train loss and validation loss, are reported in Table 2 and the validation loss is visualised in Figure 20. The figure also shows the validation losses of the individual models.

Table 2: Training results of a multi-user model. CNN (cc.) stands for combined-channel version of the CNN model.

	MLP	CNN	CNN (cc.)	LSTM-MLP	LSTM-CNN
Runtime (s)	117	440	158	179	688
Parameters	$7.2 \cdot 10^5$	$6.8 \cdot 10^6$	$1.3 \cdot 10^6$	$7.6 \cdot 10^5$	$6.8 \cdot 10^6$
Train loss	$5.7 \cdot 10^{-3}$	$4.7 \cdot 10^{-3}$	$7.3 \cdot 10^{-3}$	$2.0 \cdot 10^{-3}$	$1.6 \cdot 10^{-3}$
Val loss	$8.2 \cdot 10^{-3}$	$8.0 \cdot 10^{-3}$	$11.2 \cdot 10^{-3}$	$5.7 \cdot 10^{-3}$	$4.4 \cdot 10^{-3}$

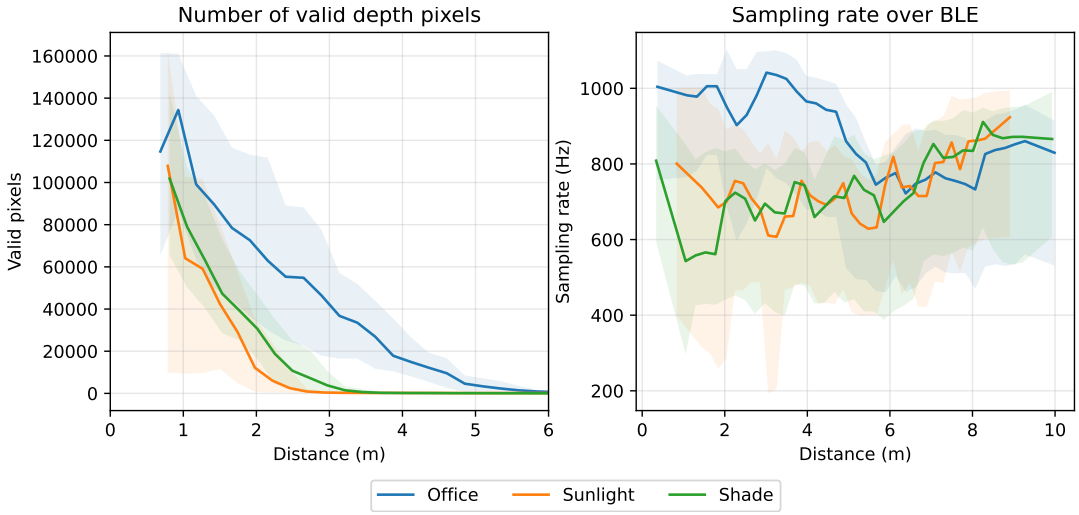

Figure 19: Performance of the used sensors in scenarios depicted in Figure 8. Left: Valid depth camera pixels within a bounding box of a detected person. Right: Received sampling rate of the sEMG signal over BLE. The curves correspond to the mean of the data, whereas the lower and the upper boundaries of the shaded area correspond to the 5% and 95% percentiles of the original data.

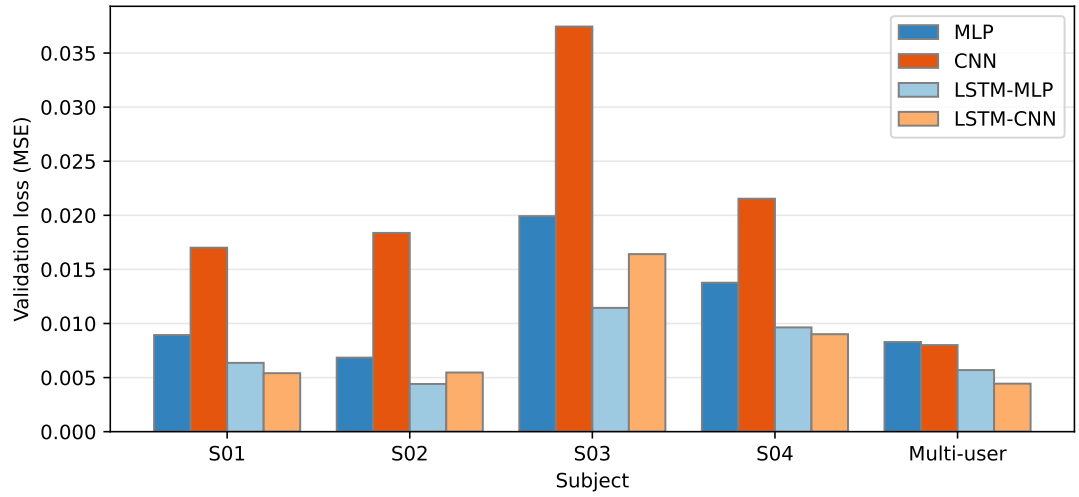

Figure 20: Offline proportional control performance of the tested models for each subject averaged over five iterations. Trained using sessions 1, 2 and 3, and tested with session 5. Multi-user model was trained using combined data from all subjects.

The effect of two design choices were evaluated, when compared to the base models. First, the effect of donning and doffing was investigated by including data recorded in the same session as the validation data when training the model. Based on a compromise between the number of parameters and the validation loss, the LSTM-MLP model was chosen as the baseline. The baseline is labelled as inter-

session in Figure 21 and the model trained on the altered dataset is labelled as intra-session. On average, including data from the same session as the validation data decreased the validation loss by 14.5 % on individual models and 3.8 % on the multi-user model.

The choice of separating the EMG channels in the convolutional layers of the CNN and LSTM-CNN models, as described in Section 3.3.2, was the second design choice investigated. Individual and multi-user models were trained with the combined-channel architecture, and the validation losses are shown in Figure 21 against the separate-channel counterparts. On average, the validation loss of the individual models decreased by 20.8 % when using the separate-channel architecture and 28.6 % smaller with the multi-user model.

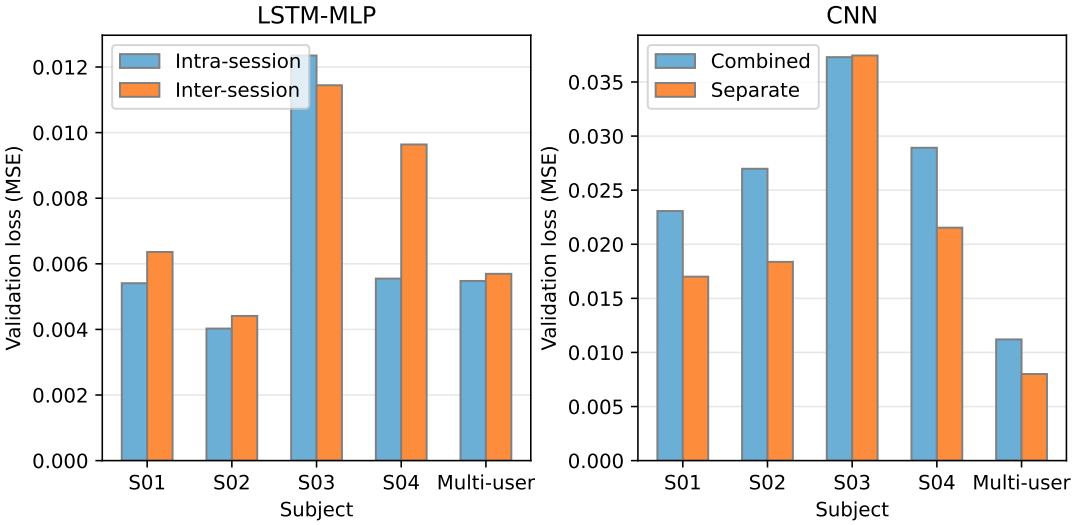

Figure 21: Left: Offline proportional control performance of LSTM-MLP models trained with intra-session and inter-session subsets of the data. Inter-session results are the same as in Figure 20, whereas intra-session models were trained using sessions 1, 2 and 4 for training, and 5 for validation. Right: Offline proportional control performance of the combined-channel variant of the CNN model and the separate-channel variant. The latter variant was used in Figure 20.

4.4 Evaluation

The proof of concept consisted of walking the robot around a predetermined route and instructing it to perform a set of actions using the proposed interface. The resulting paths are depicted in Figure 22. For visualisation purposes, the rotation and the starting point of the paths were aligned, as the odometry of the robot drifted by some amount. The drift is still visible in the figure, however, as the starting points and ending points were approximately the same in the real world, whereas in the figure, there is a slight gap between them (points A and E). On average, the

angle was corrected by 2.75° between each session, and there was a 2.83 m difference between the starting and the ending point.

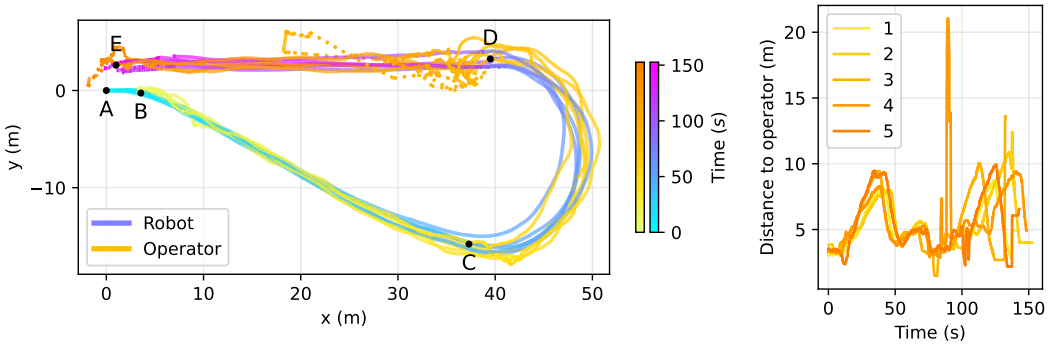

Figure 22: Left: Paths of the operator and the robot, aligned at the starting point. Right: The distance between the operator and the robot during each of the five repetitions.

Completing the sequences took 147 s on average with a standard deviation of 9.9 s. On the first relatively straight segment of the path (from point B to C in Figure 22), the average speed of the robot was 1.21 ±0.21 m/s based on the odometry of the robot. The speed of the operator was estimated using the operator detection module of the interface, which resulted in an average speed of 1.44 ±0.24 m/s on the same segment of the path.

The robot was stopped and the pictures were taken at point D in the figure. This required some additional movement from the operator which can be seen adjacent to the point. A short series of false detections of the operator can be seen between points D and E, when the tracking of the operator abruptly jumps towards E. Similarly, there is a sudden increase on the distance between the operator and the robot. The detections originated from a person, but not from the one controlling the robot.

Precision, recall and F-score were calculated for each gesture type used in the sequences. The overall performance of the interface and the robot combined is shown in Table 3. Here, all attempts at commanding the robot are included. Table 4 shows the same data, but some of the gesture attempts were changed or removed. These include attempts that failed due to problems with BLE data transmission or the robot not performing the issued task, despite receiving a correct command. In the latter case, the gestures were classified as incorrect in Table 3 and correct in Table 4. Ulnar deviation was not required for completing the sequences, and is not included in these tables.

Table 5 shows the time that was required to perform each of the gesture types. The time was measured from when the operator reached the pose of the gesture to relaxation of the pose. Usually the relaxation began when the user received feedback that the gesture recognition was successful. Otherwise, the gesture was continued until the operator concluded that the gesture would not trigger before trying again.

The total time spent performing gestures during a session was 22.7 s on average with a standard deviation of 4.1 s. On average, it spans 15.5 % of the duration of

the entire route.

Table 3: Realtime gesture detection and task completion performance, based on intended gesture vs. action realised by the robot.

	Average	Open	Fist	Extension	Flexion	Radial
Precision	0.95	0.92	1.00	0.95	1.00	0.94
Recall	0.86	1.00	0.92	0.95	0.33	0.88
F-score	0.90	0.96	0.96	0.95	0.5	0.91

Table 4: Realtime performance of the neural network alone. Misclassifications due to either poor data transmission or the robot not performing the issued command are not considered here.

	Average	Open	Fist	Extension	Flexion	Radial
Precision	0.97	0.92	1.00	0.95	1.00	1.00
Recall	0.93	1.00	1.00	0.95	0.75	0.88
F-score	0.95	0.96	1.00	0.95	0.86	0.94

Table 5: Average times in seconds for successfully triggered gestures. Measured from reaching the characteristic pose of the gesture to relaxation.

	Average	Open	Fist	Extension	Flexion	Radial
Average	1.76	1.42	1.18	1.99	2.77	1.91
SD	0.94	0.33	0.26	0.71	2.56	1.15

5 Discussion

In addition to sEMG sensors, the gForcePro+ armband includes a magnetic and intertial measurement unit (MIMU). The accelerometer and gyroscope signals are regularly used for gesture detection, whereas magnetometers are often disregarded. Magnetometer can be used to improve sensorfusion-based estimates of the device orientation. It affects mostly the heading component of the estimate, which is not generally relevant when recognising simple gestures. The direction towards which the gestures are performed could even degrade the results if sufficient variation of the heading is not included in the data.

Combining EMG and IMU signals as the input of the algorithms has yielded better results than relying only to either IMU or EMG signals alone [13,14,40]. Using IMU data, however, could complicate the data collection procedure, especially in cases where the intended user movement is less restricted than that of laboratory conditions. In these cases, unseen hand movement and gesture combinations could have a negative effect to the results. On the other hand, inclusion of IMU signals makes dynamic gestures more recognisable, given that sufficient variation is present in the data. If an IMU is used, the subjects should be moving similarly as they would when the system is used outside the laboratory.

There are advantages and disadvantages of using a sEMG armband together with a camera-based user recognition system for controlling a robot when compared to other input schemes. Gesture recognition can be achieved with purely camera-based systems and the detection of the user does not have to rely on detecting the bracelet, as was done in this thesis. Without detecting some sort of marker for the user, other initialisation methods could include a predefined gesture, such as a hand wave, which is recognised and the corresponding user is tracked for as long as a line of sight to the camera is maintained. A loss of a track would then have more adverse effects, as the user would have to reinitialise the session. The tracking could be improved by incorporating a histogram of the pixels corresponding to the user, which could be matched in future frames. More advanced approaches include deep learning-based user reidentification, as demonstrated in [87,88].

If no marker or a detected object, such as an armband, is used for user detection, the risk of wrong initialisations is larger. A simple image-based recognition of an armband does not act as a valid security system for accessing the robot, especially as it could be tricked with a picture of the armband and the occurrence of other false positives cannot be completely eradicated, but it should decrease the occurrence of unintentional initialisation, especially in crowded areas.

A key advantage of using only cameras is that the user is not required to wear any additional devices. The use of these devices can be cumbersome as they require batteries to work, which need to be recharged occasionally, and the systems used for inference might need retraining for each person for optimal performance. On the other hand, deep learning-based people detection and pose detection systems have been shown to generalise well between different people. However, the finest gestures are not easily detectable by cameras, given that the whole body of the user is in view. Especially proportional control using a hand or a wrist includes fine movements that

could be hard to reliably detect from images alone. This would limit the size of the gestures to medium and large, which can easily be wearing to the user in long term use. In addition, any breaks in the line of sight between the camera and the user would render the system unusable until the line of sight is regained.

Gesture recognition and user detection can be achieved using only an IMU armband. This was demonstrated in [22], where pointing gesture was used to identify and localise a user who wants to interact with a robot and their pointing direction was determined using the established localisation. This method also required the local trajectory of the robot. The scale of gestures recognisable with an IMU armband is similar to camera-based systems, that is, the gestures need to feature sufficient arm movement to be distinguishable. Using multiple IMUs could improve the separability, but it also increases the preparation time, as the devices need to be set up before use.

Gesture detection using an IMU require a considerably lower bandwidth for data than what EMG requires. Typically, the IMUs support sampling rates of around 100 Hz, while a lower rate is usually sufficient [101]. At maximum, the IMU requires nine channels, three components for each accelerometer, gyroscope and magnetometer outputs, but some of these are optional depending on the task. The gForcePro+ armband supports 50 Hz sampling rate for its integrated IMU, while the EMG is recorded at a rate between 500 and 1000 Hz and there are a total of eight channels in the device [93]. Other systems include even higher channel counts and sampling rates. Assuming the amount of data transmitted correlates with the energy use of the device, a device using only IMU would be able to operate longer with the same battery, thus requiring less frequent recharging.

The advantages of using a sEMG bracelet over the previously mentioned input schemes include the ability to recognise fine gestures, such as different hand poses and proportional control by bending the wrist. The device does not require a line of sight and is not affected by changes in lighting conditions, which helps to make the system more consistent in various environments than a purely camera-based system. Even when using the armband together with a camera-based user detection, the gestures do not require the camera for input, making the system more reliable. The armband is worn by the intended user, which makes the system less susceptible to detecting wrong person as the user and practically immune to recognising gestures from other people than the one wearing the armband.

The data from EMG and IMU devices are complementary in the sense that the EMG signals include information about the fine movements mostly from below the elbow, while signals from an IMU include mostly larger scale information about the pose of the arm [14]. If either of these devices is worn on the arm with an intent to recognise gestures, there is a good reason for including the other one as well.

However, including additional modalities as an input to a machine learning model, increases the models tendency to overfit to the data, which could be a valid reason to not include either of these modalities. EMG is especially susceptible to overfitting and various studies have tried to address the problems with intersession and inter-subject generalisation. So far, no universal solution has been reported that would be able to reliably classify gestures regardless of the session and the subject.

The gesture detection performance based on the Ninapro DB2 dataset improved when the sampling rate was increased, as shown in Figure 15. However, when the ADC resolution was increased, the performance first improved, but then started to decrease. The initial improvement can be explained by the increased amount of information present in the signal. In this sense, the performance of the model should increase further with the resolution, but as seen in the figure, this is not the case and the performance peaked at a resolution of 8 bits. It is suspected that this is due to a filtering effect of the limited resolution, which acts as a low-pass filter for the signal. Typically, a low-pass filter of 500 Hz is applied for sEMG signals to reduce the amount of noise present [38]. Here, an antialiasing low-pass filter was applied when resampling the signals.

Based on the analysis on Ninapro DB2 dataset, the gForcePro+ armband is a more potential option for sEMG-based gesture detection, when compared to the other similar device, the Myo armband. Both available operating modes of gForcePro+, 12 bits at 500 Hz, and 8 bits at 1 kHz perform better in this analysis than the 8 bits at 200 Hz mode available to Myo. Furthermore, the 1 kHz mode performs better than the 500 Hz mode and for this reason, the 1 kHz operating mode is recommended for the gForcePro+ armband. However, this is merely a theoretical result, as the signals used for this analysis are not recorded with the device itself. Further studies using the device would be required to confirm these results. Nevertheless, the 8 bits at 1 kHz operation mode was used for the measurements with gForcePro+.

Obtaining sEMG-based gesture detection data is time consuming and public datasets are not available for each device. Transfer learning between devices, i.e., inter-device transfer learning, could potentially be used to address this problem, assuming that the signals can be converted to match the target device well enough. The Ninapro DB2 dataset suits this application well, as the original sampling rate is high enough to be downsampled to match most devices. Matching the signal characteristics between devices would require other solutions, such as system identification, domain randomisation or generative neural networks, such as GANs. Some of these methods, however, also require considerable amounts of data from the target domain, thus reducing the potential benefits. This is why a direct mapping between the signals of the devices would seem most promising, as the final adjustment could be applied with transfer learning. In practice, inter-device transfer learning would be an extension to inter-subject transfer learning, as the subjects would most likely be different between the datasets.

The Ninapro DB2 dataset contains 40 subjects, which is sufficient to confirm the viability of the gesture detection algorithm for within-session use. For an use case in HRI, however, it is not feasible, or at least not very convenient, to collect a new set of data and retrain the neural network every time before use. For this reason, various options for overcoming this limitation were introduced and experimented with.

Conceptually the simplest case is to collect a dataset which includes donning and doffing between sessions. Ideally, a neural network trained from such dataset would be immune to the temporal variation of the sEMG signal as well as the variation in the electrode placements. However, without sufficient variation in the data and when using a larger model, there is a risk that the model would simply learn to recognise

the different configurations leading to poor generalisability.

The applicability of domain adaptation for sEMG-based gesture detection was studied in [69]. In addition to the explored options, utilising CycleGANs [102] to perform the adaptation of sEMG signals could be investigated. CycleGANs are an extension of GANs designed for, but not limited to, tasks such as style transfer and photo enhancement. CycleGANs are used to transfer the input data from a source domain to a target domain and back, thus ensuring the preservation of the characteristic features of the data. By using CycleGANs, a large dataset could be transformed to the domain of a smaller dataset. As with GANs, however, CycleGANs can require large datasets to be useful.

Although the two outside scenarios, depicted in Figure 8, resulted in similar looking BLE performances, as shown in Figure 19, the connection was lost multiple times when performing the measurements in the sunlight scenario. This resulted in shorter maximum distance that was reached before the connection was lost, when compared to the other scenarios. It is suspected that this was due to a nearby transformer of an office building, as similar performance was not observed at the other side of the same building.

The reason behind the drop in the received sampling rate is not entirely clear. The theoretical range should be long enough for this use case, as the reported radio power of the gForcePro+ is 4 dBm [93]. Using an isotropic antenna for transmission and a receiving antenna with a minimum receiver sensitivity of -70 dBm, the theoretical range should reach 23 m, as calculated by Equation 6, whereas with a sensitivity of -90 dBm available to modern chips, the range should be as high as 100 m [103]. This is clearly not the case here, as the optimal range extends only up to 4 m, and the connection was usually cut completely after 10 m. It is possible that there are additional interference sources present in these environments that would degrade the signal, which were not identified here.

$$d = 10^{(path\ loss - 40)/25} \tag{6}$$

The distance estimate was only accepted when the valid pixel count within the bounding box of a detected person exceeds 10^3 pixels, corresponding to roughly 5 meters indoors and 2-2.5 m outdoors. This is not a limit for the system, however, as the distance estimate range was extended by using an estimator based on the height of the bounding box. As seen in Figure 17, the regression curve fits well to the data, while being close to the theoretical result. Based on this data, it is not necessarily required to calibrate the estimator in a separate session, and simply providing the height of the person should suffice. Alternatively, the height of the person could be calibrated automatically in a short session.

The caveat of using the bounding box-based distance estimation is that the entire person has to be visible for the estimates to be reliable. As seen in Figure 18, the distance cannot be estimated closer than a certain limit, that depends on the height of the person. For a 1.72 m tall person, for example, the theoretical limit with the given camera is roughly 2.0 m. Occlusions of the person is also a problem, which were not addressed here and are considered a limitation of this method. However, on

closer distances the depth camera is used instead, which does not suffer from the same problem.

Based on these results, the direct connection between the armband and the robot would be usable up to a distance of 4 meters between the operator and the robot in a low interference environment. This is where the received sampling rate begins to drop below the nominal sampling rate of 1 kHz and the reliability of detecting the gestures cannot be guaranteed. As the system was intended to work in less ideal circumstances, a laptop was introduced for the complete system evaluation. The sEMG armband was connected to the laptop with BLE, from which the data was forwarded to the robot. In the future, the laptop should be replaced with a smartphone, or a similar small-sized device carried by the operator, that is able to communicate with the armband and forward the received data to the robot through more reliable means of communication. Depending on the environment and available technology, at least Wi-Fi and cellular networks could be considered.

A finite-state machine is a standard solution for programming the behaviour of a robot or any similar agent and it proved to work here as well. However, some changes for the implemented system would be recommended. When the robot was commanded to follow the operator, the speed of the robot was always separately increased to maximum, which required two additional gestures to be performed. Instead, the speed should be a variable within the follow state, which could be varied by the extension and flexion gestures. When leaving or entering the follow state using the fist and hand open gestures, the speed would be stored and applied automatically, thus removing the redundant gestures and reducing the time required to begin following.

From the neural network architectures evaluated in this work, the multi-user LSTM-CNN performed the best with an MSE of $4.4 \cdot 10^{-3}$. However, the size of the model was considerably larger than the size of LSTM-MLP: almost by a factor of 10. As the computational resources of Jetson Xavier are limited, the LSTM-MLP model was chosen instead, as it achieved the second best results with an MSE of $5.7 \cdot 10^{-3}$.

The time required for training each of the models was quite short, ranging roughly between 2.0 and 11.5 minutes for the multi-user models when trained on an Titan RTX GPU. The low requirements in terms of training time can be attributed to the relatively small models and data sets. Fine-tuning an already trained model usually takes less time than the original training, meaning that a gesture detection system like this could be updated in a reasonable time even when using less powerful hardware.

The effect of donning and doffing was present, but the results varied between subjects as previously shown in Figure 21. For the first two subjects, the effect was a relatively small 11.8 % decrease on average, whereas for the fourth subject, changing one of the sets with intra-session data reduced the MSE loss by 42.4 %. On the other hand, the MSE loss increased by 7.9 % for the third subject. The results for the third and fourth subject indicate that changing the data of one of the three sessions used for training can affect the results significantly. Three sessions might not be enough to train a the individual models reliably. The effect on multi-user models, on the other hand, was relatively small, only 3.8 %. Using data from multiple subjects introduces

more variability to the data, which can overlap with the variability resulting from donning and doffing, thus allowing the models to generalise better.

Overall, the regression error for the third subject stands out from the rest of the subjects. Interestingly, the third subject had the lowest estimated standard deviation in the placement of the armband: 0.24 cm whereas the average for the other subjects was 1.07 cm. As this cannot explain the results, it is instead suspected that the subject performed the gestures with a lower level of muscle activation than the others, which could negatively affect the performance of the neural networks.

The individual separate-channel CNNs, which were trained with 1/4 of the data of the multi-user models, performed better on average than the individual combined-channel variants of the CNN. This is noteworthy, as the individual CNN models performed considerably worse when compared to other architectures, but the multi-user CNN outperformed the multi-user MLP as shown in Figure 20. The assumed reason for this is the size of the training set, which could be too small to train the individual CNNs. However, such effect cannot be seen between separate-channel and combined-channel CNNs. The change in validation loss between multi-user variants is on par with the change in individual variants, as seen in Figure 21. It could be assumed, that when the number of parameters is increased from $1.3 \cdot 10^6$ to $6.8 \cdot 10^6$, the performance of the individual models would decrease as a result of overfitting. As this is not the case, these results could indicate that the separate-channel variants help achieving better results on sEMG-based gesture regression without causing the model to overfit.

While using the separate-channel architecture helped reducing the validation loss, concatenating the outputs of each channel increased the layer size considerably. For a more equal comparison, the number of parameters should be matched more closely. There might also exist a better balance between separate and combined-channel architectures, as mixtures between these models were not investigated.

Additionally, the separation of IMU channels in the CNN architecture should be investigated as a way to improve the results further. The potential for the improvements with separating IMU channels is smaller, however, as they include three accelerometer and three gyroscope channels, whereas with EMG, there are eight channels featuring relatively similar signals.

The average time it took for the gestures to trigger was large because there was uncertainty whether the gesture would trigger or not as sometimes the gesture triggered noticeably late assumably due to problems with the BLE connection. Furthermore, it was sometimes unclear whether the gesture was already triggered, due to the weak form of feedback and sometimes the proportional output from the neural network was too small to trigger the gesture.

The gesture times also include the reaction time of the operator. Assuming a reaction time of 250 ms, the average time from onset of a gesture to triggering reduces to 1.51 s. This is still not within the recommended limit of 300 ms, but for instructing a semi-autonomous robot, near-instantaneous responses might not be necessary. The reliability of the system was prioritised, and it could be possible to reduce the required time to trigger a gesture without affecting the precision too much. Additionally, lowering the threshold for triggering a gesture could improve

the recall of the detections.

The total time spent performing gestures and thus, the time spent actively controlling the robot, was 15.5 % of the duration of the entire route. With the previously mentioned changes, i.e., fixing the connection issues and reducing the triggering time, this could be reduced even further. When compared to a hand-held controller, which requires constant input from the user, this is a significant improvement. The time required to perform gestures naturally depends on the route and tasks being performed. With longer walking distances, the relative time required to perform gestures should decrease and vice versa.

Similarly, other proposed sEMG-based interfaces [15–17] for mobile robots also require constant input from the user. If the gesture-based input is required to be constant, the advantage over a hand-held controller is smaller and the user fatigues faster. Muscle fatigue affects the frequency spectrum of EMG [24], which, if not addressed properly, can deteriorate gesture detection [18].

Finally, the following behaviour of the robot needs to be addressed. While the following was consistent and reliable, the current system instructs the robot to walk directly towards the operator. This is not always desirable, as the robot now has a tendency to cut corners regardless of the path taken by the operator. Instead, the robot should follow the operator's path more closely, which in turn could cause the operator to leave the field of view of the robot. This in turn could be alleviated by either having the camera of the robot turn independently towards the operator, or by tracking the position of the operator by using additional devices such as augmented reality glasses.

6 Conclusion

This thesis proposed and validated the first iteration of a novel HRI interface for controlling mobile robots. In the previous chapter, the interface was compared to similar solutions and it was found to be capable of supporting a broader set of instructions than other sEMG-based interfaces. The performance of the gesture detection neural network was on par with that of other neural networks from the literature.

Overall, the LSTM-CNN architecture achieved the best regression results from the architectures used here, when the data from all the subjects were used to train a multi-user model. Separating the EMG channels proved to improve the results, but with a relatively high cost in computing time, which is why the LSTM-MLP model was used in the proof of concept. While donning and doffing is known to deteriorate the inference of gestures, the effect quantified here was quite small, and with real-time haptic feedback, the proof of concept could be conducted successfully without retraining or recalibrating the system. The camera-based operator detection enabled the proof of concept, and even though there are room for improvements, the consequences of the few operator detection errors encountered were minimised with the use of gesture control.

The gesture detection model was selected from a set of candidate models by comparing their MSE loss on the validation set. For the operator detection, a object detection neural network was retrained to enable detecting both people and the sEMG armbands. The proof of concept for the complete system was analysed thoroughly. F-scores of the used gestures were calculated for both the gestures themselves and as a part of the system. The time required to perform the gestures was measured as well as the time to complete the whole course. A set of calibrations for the different modules of the interface were performed prior to the the proof of concept.

Hand-crafted features are a well-established tool for extracting information from EMG signals. With the recent advancements in neural networks, more complex models have been proposed for sEMG-based gesture detection as well, surpassing the traditional methods. The model complexity, however, introduces increased computational cost and requirements for larger datasets, which is why the diminishing returns of the more complex models can not always be justified.

The field of people detection and tracking, on the other hand, enjoys a richer selection of larger datasets, which can be used to train more specialised models. People re-identification and pose estimation are among the tasks that seem potential for an HRI interface, such as the one presented in this thesis. While visual gesture detection has yielded promising results, the current limitations keep it from being applied to controlling mobile robots.

The modalities for the interface were set to include an sEMG armband in addition to visual and depth cameras. The feasibility of the system was demonstrated and it was concluded that using an sEMG armband and the proposed interface enables users to interact with robots without many of the limitations with visual or auditory methods or without having to operate the hand-held controller. Combining a semi-autonomous user-following robot with gesture control reduces the time required to

actively control the robot.

The main limitation of the work presented here is that the proof of concept was performed with only one subject. This was considered adequate, as the gesture detection was already validated with multiple subjects and other aspects of the interface are less person-specific. Nevertheless, additional validation for the complete interface is required. The validation for the operator detection was minimal, as it was primarily considered an enabler for this work. This leads to the next steps, which include improving the operator detection system with either person reidentification, augmented reality glasses, or both. This enables a richer set of available tasks, which can already be supported with the current gesture detection system. The set of gestures could be increased further while their required activation time should be decreased.

User-following robots and sEMG-based gesture detection are both promising technologies, the potential of which is only increased when combined together with a multimodal interactive interface.

References

[1] A. B. Hostetter, "When do gestures communicate? a meta-analysis." *Psychological bulletin*, vol. 137, no. 2, p. 297, 2011.

[2] C. Obermeier, T. Dolk, and T. C. Gunter, "The benefit of gestures during communication: Evidence from hearing and hearing-impaired individuals," *Cortex*, vol. 48, no. 7, pp. 857–870, 2012.

[3] J. Liu, X. Sheng, D. Zhang, J. He, and X. Zhu, "Reduced daily recalibration of myoelectric prosthesis classifiers based on domain adaptation," *IEEE journal of biomedical and health informatics*, vol. 20, no. 1, pp. 166–176, 2014.

[4] U. Côté-Allard, C. L. Fall, A. Drouin, A. Campeau-Lecours, C. Gosselin, K. Glette, F. Laviolette, and B. Gosselin, "Deep learning for electromyographic hand gesture signal classification using transfer learning," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 27, no. 4, pp. 760–771, 2019.

[5] A. Phinyomark, P. Phukpattaranont, and C. Limsakul, "Feature reduction and selection for emg signal classification," *Expert systems with applications*, vol. 39, no. 8, pp. 7420–7431, 2012.

[6] A. Phinyomark, R. N Khushaba, and E. Scheme, "Feature extraction and selection for myoelectric control based on wearable emg sensors," *Sensors*, vol. 18, no. 5, p. 1615, 2018.

[7] N. K. Karnam, S. R. Dubey, A. C. Turlapaty, and B. Gokaraju, "Emghandnet: A hybrid cnn and bi-lstm architecture for hand activity classification using surface emg signals," *Biocybernetics and Biomedical Engineering*, vol. 42, no. 1, pp. 325–340, 2022.

[8] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, and W. Geng, "A novel attention-based hybrid cnn-rnn architecture for semg-based gesture recognition," *PloS one*, vol. 13, no. 10, p. e0206049, 2018.

[9] I. Vujaklija, V. Shalchyan, E. N. Kamavuako, N. Jiang, H. R. Marateb, and D. Farina, "Online mapping of emg signals into kinematics by autoencoding," *Journal of neuroengineering and rehabilitation*, vol. 15, no. 1, pp. 1–9, 2018.

[10] R. Meattini, M. Nowak, C. Melchiorri, and C. Castellini, "Automated instability detection for interactive myocontrol of prosthetic hands," *Frontiers in neurorobotics*, vol. 13, p. 68, 2019.

[11] R. A. Bos, K. Nizamis, B. F. Koopman, J. L. Herder, M. Sartori, and D. H. Plettenburg, "A case study with symbihand: an semg-controlled electrohydraulic hand orthosis for individuals with duchenne muscular dystrophy," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 28, no. 1, pp. 258–266, 2019.

[12] M. Lyu, W.-H. Chen, X. Ding, J. Wang, Z. Pei, and B. Zhang, "Development of an emg-controlled knee exoskeleton to assist home rehabilitation in a game context," *Frontiers in neurorobotics*, vol. 13, p. 67, 2019.

[13] M. Georgi, C. Amma, and T. Schultz, "Recognizing hand and finger gestures with imu based motion and emg based muscle activity sensing." in *Biosignals*. Citeseer, 2015, pp. 99–108.

[14] J. Wu, L. Sun, and R. Jafari, "A wearable system for recognizing american sign language in real-time using imu and surface emg sensors," *IEEE journal of biomedical and health informatics*, vol. 20, no. 5, pp. 1281–1290, 2016.

[15] S. Bisi, L. De Luca, B. Shrestha, Z. Yang, and V. Gandhi, "Development of an emg-controlled mobile robot," *Robotics*, vol. 7, no. 3, p. 36, 2018.

[16] E. Mohamed, K. H. Tantawi, A. Pemberton, N. Pickard, M. Dyer, E. Hickman, K. Thompson, E. Kaplanoglu, and A. Nasab, "Real time gesture-controlled mobile robot using a myo armband," in *Proceedings of the International Conference on Industrial Engineering and Operations Management*, vol. 59, 2020, pp. 2432–2437.

[17] T. Chu, A. Chua, and E. L. Secco, "A wearable myo gesture armband controlling sphero bb-8 robot," *HighTech and Innovation Journal*, vol. 1, no. 4, pp. 179–186, 2020.

[18] P. K. Artemiadis and K. J. Kyriakopoulos, "An emg-based robot control scheme robust to time-varying emg signal features," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 3, pp. 582–588, 2010.

[19] S. Waldherr, R. Romero, and S. Thrun, "A gesture based interface for human-robot interaction," *Autonomous Robots*, vol. 9, no. 2, pp. 151–173, 2000.

[20] M. Tölgyessy, M. Dekan, F. Duchoň, J. Rodina, P. Hubinskỳ, and L. Chovanec, "Foundations of visual linear human–robot interaction via pointing gesture navigation," *International Journal of Social Robotics*, vol. 9, no. 4, pp. 509–523, 2017.

[21] J. Montesdeoca, J. M. Toibero, J. Jordan, A. Zell, and R. Carelli, "Person-following controller with socially acceptable robot motion," *Robotics and Autonomous Systems*, vol. 153, p. 104075, 2022.

[22] B. Gromov, L. M. Gambardella, and A. Giusti, "Robot identification and localization with pointing gestures," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3921–3928.

[23] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, TaoXie, J. Fang, imyhxy, K. Michael, Lorna, A. V, D. Montes, J. Nadar, Laughing, tkianai, yxNONG, P. Skalski, Z. Wang, A. Hogan, C. Fati, L. Mammana, AlexWang1900, D. Patel, D. Yiwei, F. You, J. Hajek, L. Diaconu,

and M. T. Minh, "ultralytics/yolov5: v6.1 - TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference," Feb. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6222936

[24] W. Herzog and B. M. Nigg, *Biomechanics of the Musculo-skeletal System*. John Wiley & Sons, 2006.

[25] C. J. De Luca, A. Adam, R. Wotiz, L. D. Gilmore, and S. H. Nawab, "Decomposition of surface emg signals," *Journal of neurophysiology*, vol. 96, no. 3, pp. 1646–1657, 2006.

[26] C. D. Katsis, T. P. Exarchos, C. Papaloukas, Y. Goletsis, D. I. Fotiadis, and I. Sarmas, "A two-stage method for muap classification based on emg decomposition," *Computers in Biology and Medicine*, vol. 37, no. 9, pp. 1232–1240, 2007.

[27] A. Holobar, M. A. Minetto, A. Botter, F. Negro, and D. Farina, "Experimental analysis of accuracy in the identification of motor unit spike trains from high-density surface emg," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 3, pp. 221–229, 2010.

[28] C. J. De Luca, S.-S. Chang, S. H. Roy, J. C. Kline, and S. H. Nawab, "Decomposition of surface emg signals from cyclic dynamic contractions," *Journal of neurophysiology*, vol. 113, no. 6, pp. 1941–1951, 2015.

[29] E. A. Clancy and N. Hogan, "Probability density of the surface electromyogram and its relation to amplitude detectors," *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 6, pp. 730–739, 1999.

[30] M. Simão, P. Neto, and O. Gibaru, "Emg-based online classification of gestures with recurrent neural networks," *Pattern Recognition Letters*, vol. 128, pp. 45–51, 2019.

[31] J. Kilby, K. Prasad, and G. Mawston, "Multi-channel surface electromyography electrodes: a review," *IEEE Sensors Journal*, vol. 16, no. 14, pp. 5510–5519, 2016.

[32] C.-F. Chi, Y.-C. Shih, and W.-L. Chen, "Effect of cold immersion on grip force, emg, and thermal discomfort," *International Journal of Industrial Ergonomics*, vol. 42, no. 1, pp. 113–121, 2012.

[33] N. Sezgin, "Analysis of emg signals in aggressive and normal activities by using higher-order spectra," *The Scientific World Journal*, vol. 2012, 2012.

[34] M. Cracchiolo, G. Valle, F. Petrini, I. Strauss, G. Granata, T. Stieglitz, P. M. Rossini, S. Raspopovic, A. Mazzoni, and S. Micera, "Decoding of grasping tasks from intraneural recordings in trans-radial amputee," *Journal of Neural Engineering*, vol. 17, no. 2, p. 026034, 2020.

[35] F. Lotti, F. Ranieri, G. Vadalà, L. Zollo, and G. Di Pino, "Invasive intraneural interfaces: foreign body reaction issues," *Frontiers in neuroscience*, vol. 11, p. 497, 2017.

[36] G. Li, Y. Li, Z. Zhang, Y. Geng, and R. Zhou, "Selection of sampling rate for emg pattern recognition based prosthesis control," in *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*. IEEE, 2010, pp. 5058–5061.

[37] J. C. Ives and J. K. Wigglesworth, "Sampling rate effects on surface emg timing and amplitude measures," *Clinical biomechanics*, vol. 18, no. 6, pp. 543–552, 2003.

[38] M. Simão, N. Mendes, O. Gibaru, and P. Neto, "A review on electromyography decoding and pattern recognition for human-machine interaction," *Ieee Access*, vol. 7, pp. 39 564–39 582, 2019.

[39] A. W. Wilson, Y. G. Losier, P. A. Parker, and D. F. Lovely, "A bus-based smart myoelectric electrode/amplifier—system requirements," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 10, pp. 3290–3299, 2011.

[40] W. Wei, Q. Dai, Y. Wong, Y. Hu, M. Kankanhalli, and W. Geng, "Surface-electromyography-based gesture recognition by multi-view deep learning," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 10, pp. 2964–2973, 2019.

[41] W. Jiang and Z. Yin, "Human activity recognition using wearable sensors by deep convolutional neural networks," in *Proceedings of the 23rd ACM international conference on Multimedia*, 2015, pp. 1307–1310.

[42] K. Englehart and B. Hudgins, "A robust, real-time control scheme for multi-function myoelectric control," *IEEE transactions on biomedical engineering*, vol. 50, no. 7, pp. 848–854, 2003.

[43] R. Menon, G. Di Caterina, H. Lakany, L. Petropoulakis, B. A. Conway, and J. J. Soraghan, "Study on interaction between temporal and spatial information in classification of emg signals for myoelectric prostheses," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 10, pp. 1832–1842, 2017.

[44] M. Ison and P. Artemiadis, "The role of muscle synergies in myoelectric control: trends and challenges for simultaneous multifunction control," *Journal of neural engineering*, vol. 11, no. 5, p. 051001, 2014.

[45] M. Ison, I. Vujaklija, B. Whitsell, D. Farina, and P. Artemiadis, "High-density electromyography and motor skill learning for robust long-term control of a 7-dof robot arm," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 4, pp. 424–433, 2015.

[46] N. Jiang, K. B. Englehart, and P. A. Parker, "Extracting simultaneous and proportional neural control information for multiple-dof prostheses from the surface electromyographic signal," *IEEE transactions on Biomedical Engineering*, vol. 56, no. 4, pp. 1070–1080, 2008.

[47] E. Stevens, L. Antiga, and T. Viehmann, *Deep learning with PyTorch.* Manning Publications, 2020.

[48] S. Kiranyaz, T. Ince, R. Hamila, and M. Gabbouj, "Convolutional neural networks for patient-specific ecg classification," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC).* IEEE, 2015, pp. 2608–2611.

[49] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures," *Neural computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[50] M. Atzori and H. Müller, "The ninapro database: a resource for semg naturally controlled robotic hand prosthetics," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC).* IEEE, 2015, pp. 7151–7154.

[51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[52] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang *et al.*, "A comparative study on transformer vs rnn in speech applications," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU).* IEEE, 2019, pp. 449–456.

[53] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[54] B. Wu, C. Xu, X. Dai, A. Wan, P. Zhang, Z. Yan, M. Tomizuka, J. Gonzalez, K. Keutzer, and P. Vajda, "Visual transformers: Token-based image representation and processing for computer vision," *arXiv preprint arXiv:2006.03677*, 2020.

[55] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision.* Springer, 2020, pp. 213–229.

[56] G. Yan, S. Liang, Y. Zhang, and F. Liu, "Fusing transformer model with temporal features for ecg heartbeat classification," in *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).* IEEE, 2019, pp. 898–905.

[57] A. D'Eusanio, A. Simoni, S. Pini, G. Borghi, R. Vezzani, and R. Cucchiara, "A transformer-based network for dynamic hand gesture recognition," in *2020 International Conference on 3D Vision (3DV)*. IEEE, 2020, pp. 623–632.

[58] E. Rahimian, S. Zabihi, A. Asif, D. Farina, S. F. Atashzar, and A. Mohammadi, "Temgnet: Deep transformer-based decoding of upperlimb semg for hand gestures recognition," *arXiv preprint arXiv:2109.12379*, 2021.

[59] F. Chollet, *Deep learning with Python*, 2nd ed. Manning Publications, 2021.

[60] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *arXiv preprint arXiv:2101.01169*, 2021.

[61] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[62] J.-B. Cordonnier, A. Loukas, and M. Jaggi, "On the relationship between self-attention and convolutional layers," *arXiv preprint arXiv:1911.03584*, 2019.

[63] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[64] E. Lashgari, D. Liang, and U. Maoz, "Data augmentation for deep-learning-based electroencephalography," *Journal of Neuroscience Methods*, p. 108885, 2020.

[65] C. Sun, J. Fan, C. Chen, W. Li, and W. Chen, "A two-stage neural network for sleep stage classification based on feature learning, sequence learning, and data augmentation," *IEEE Access*, vol. 7, pp. 109 386–109 397, 2019.

[66] A. Sakai, Y. Minoda, and K. Morikawa, "Data augmentation methods for machine-learning-based classification of bio-signals," in *2017 10th Biomedical Engineering International Conference (BMEiCON)*. IEEE, 2017, pp. 1–4.

[67] P. Kaczmarek, T. Mańkowski, and J. Tomczyński, "putemg—a surface electromyography hand gesture recognition dataset," *Sensors*, vol. 19, no. 16, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/16/3548

[68] P. Tsinganos, B. Cornelis, J. Cornelis, B. Jansen, and A. Skodras, "Data augmentation of surface electromyography for hand gesture recognition," *Sensors*, vol. 20, no. 17, p. 4892, 2020.

[69] I. Ketykó, F. Kovács, and K. Z. Varga, "Domain adaptation for semg-based gesture recognition with recurrent neural networks," in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–7.

[70] J. A. Graham and M. Argyle, "A cross-cultural study of the communication of extra-verbal meaning by gestures (1)," *International Journal of Psychology*, vol. 10, no. 1, pp. 57–67, 1975.

[71] J. Nagi, F. Ducatelle, G. A. Di Caro, D. Cireşan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, and L. M. Gambardella, "Max-pooling convolutional neural networks for vision-based hand gesture recognition," in *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*. IEEE, 2011, pp. 342–347.

[72] L. Lamberti and F. Camastra, "Real-time hand gesture recognition using a color glove," in *International Conference on Image Analysis and Processing*. Springer, 2011, pp. 365–373.

[73] C. Oz and M. C. Leu, "American sign language word recognition with a sensory glove using artificial neural networks," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 7, pp. 1204–1213, 2011.

[74] G. Benitez-Garcia, L. Prudente-Tixteco, L. C. Castro-Madrid, R. Toscano-Medina, J. Olivares-Mercado, G. Sanchez-Perez, and L. J. G. Villalba, "Improving real-time hand gesture recognition with semantic segmentation," *Sensors*, vol. 21, no. 2, p. 356, 2021.

[75] E. Ceolini, C. Frenkel, S. B. Shrestha, G. Taverni, L. Khacef, M. Payvand, and E. Donati, "Hand-gesture recognition based on emg and event-based camera sensor fusion: A benchmark in neuromorphic computing," *Frontiers in Neuroscience*, vol. 14, p. 637, 2020.

[76] J.-Q. Yang, R. Wang, Y. Ren, J.-Y. Mao, Z.-P. Wang, Y. Zhou, and S.-T. Han, "Neuromorphic engineering: From biological to spike-based hardware nervous systems," *Advanced Materials*, vol. 32, no. 52, p. 2003610, 2020.

[77] Leap Motion, Inc., "Leap Motion Controller Datasheet," https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf, 2020, [Cited 11.7.2022].

[78] L. E. Potter, J. Araullo, and L. Carter, "The leap motion controller: a view on sign language," in *Proceedings of the 25th Australian computer-human interaction conference: augmentation, application, innovation, collaboration*, 2013, pp. 175–178.

[79] D. Avola, M. Bernardi, L. Cinque, G. L. Foresti, and C. Massaroni, "Exploiting recurrent neural networks and leap motion controller for the recognition of sign language and semaphoric hand gestures," *IEEE Transactions on Multimedia*, vol. 21, no. 1, pp. 234–245, 2018.

[80] A. Vaitkevičius, M. Taroza, T. Blažauskas, R. Damaševičius, R. Maskeliūnas, and M. Woźniak, "Recognition of american sign language gestures in a virtual reality using leap motion," *Applied Sciences*, vol. 9, no. 3, p. 445, 2019.

[81] Q. Yang, W. Ding, X. Zhou, D. Zhao, and S. Yan, "Leap motion hand gesture recognition based on deep neural network," in *2020 Chinese Control And Decision Conference (CCDC)*. IEEE, 2020, pp. 2089–2093.

[82] G. Marin, F. Dominio, and P. Zanuttigh, "Hand gesture recognition with leap motion and kinect devices," in *2014 IEEE International conference on image processing (ICIP)*. IEEE, 2014, pp. 1565–1569.

[83] Q. Gao, Y. Chen, Z. Ju, and Y. Liang, "Dynamic hand gesture recognition based on 3d hand pose estimation for human-robot interaction," *IEEE Sensors Journal*, 2021.

[84] M. A. Goodrich, A. C. Schultz *et al.*, "Human–robot interaction: a survey," *Foundations and Trends® in Human–Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2008.

[85] Y. Morales, T. Miyashita, and N. Hagita, "Social robotic wheelchair centered on passenger and pedestrian comfort," *Robotics and Autonomous Systems*, vol. 87, pp. 355–362, 2017.

[86] E. Bosina and U. Weidmann, "Estimating pedestrian speed using aggregated literature data," *Physica A: Statistical Mechanics and its Applications*, vol. 468, pp. 1–29, 2017.

[87] K. Koide, J. Miura, and E. Menegatti, "Monocular person tracking and identification with on-line deep feature selection for person following robots," *Robotics and Autonomous Systems*, vol. 124, p. 103348, 2020.

[88] C. Follini, V. Magnago, K. Freitag, M. Terzer, C. Marcher, M. Riedl, A. Giusti, and D. T. Matt, "Bim-integrated collaborative robotics for application in building construction and maintenance," *Robotics*, vol. 10, no. 1, p. 2, 2020.

[89] B. Gromov, G. Abbate, L. M. Gambardella, and A. Giusti, "Proximity human-robot interaction using pointing gestures and a wrist-mounted imu," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8084–8091.

[90] Boston Dynamics, "Spot | Boston Dynamics," https://www.bostondynamics.com/spot, 2022, [Cited 6.10.2022].

[91] ——, "Spot SDK," https://dev.bostondynamics.com/readme, 2022, [Cited 9.3.2022].

[92] Z. Zhang, K. Yang, J. Qian, and L. Zhang, "Real-time surface emg pattern recognition for hand gestures based on an artificial neural network," *Sensors*, vol. 19, no. 14, p. 3170, 2019.

[93] OYMotion, "gForce EMG Armband User Guide," https://oymotion.github.io/assets/downloads/gForce-EMG-ARMBAND-User-Guide-202108.pdf, 2021, [Cited 9.3.2022].

[94] L. Zhu, X. Geng, Z. Li, and C. Liu, "Improving yolov5 with attention mechanism for detecting boulders from planetary images," *Remote Sensing*, vol. 13, no. 18, p. 3776, 2021.

[95] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[96] Delsys Incorporated, "Trigno® Wireless Biofeedback System User's Guide," https://www.delsys.com/downloads/USERSGUIDE/trigno/wireless-biofeedback-system.pdf, 2021, [Cited 20.4.2022].

[97] S. Pizzolato, L. Tagliapietra, M. Cognolato, M. Reggiani, H. Müller, and M. Atzori, "Comparison of six electromyography acquisition setups on hand movement classification tasks," *PloS one*, vol. 12, no. 10, p. e0186132, 2017.

[98] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[99] H. F. Hassan, S. J. Abou-Loukh, and I. K. Ibraheem, "Teleoperated robotic arm movement using electromyography signal with wearable myo armband," *Journal of King Saud University-Engineering Sciences*, vol. 32, no. 6, pp. 378–387, 2020.

[100] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "Mediapipe hands: On-device real-time hand tracking," *arXiv preprint arXiv:2006.10214*, 2020.

[101] L. Zhou, E. Fischer, C. Tunca, C. M. Brahms, C. Ersoy, U. Granacher, and B. Arnrich, "How we found our imu: Guidelines to imu selection and a comparison of seven imus for pervasive healthcare applications," *Sensors*, vol. 20, no. 15, p. 4090, 2020.

[102] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[103] R. Heydon and N. Hunn, "Bluetooth low energy," *CSR Presentation, Bluetooth SIG https://www. bluetooth. org/DocMan/handlers/DownloadDoc. ashx*, 2012.