TEL AVIV UNIVERSITY
The Iby and Aladar Fleischman Faculty of Engineering
The Zandman-Slaner School of Graduate Studies

LEARNING HUMAN INTENTION RECOGNITION THROUGH NATURAL MOTION FOR INTUITIVE HUMAN-ROBOT INTERACTION

A thesis submitted toward the degree of
Master of Science in Engineering

by

Nadav D. Kahanowich

March 2022

TEL AVIV UNIVERSITY
The Iby and Aladar Fleischman Faculty of Engineering
The Zandman-Slaner School of Graduate Studies

LEARNING HUMAN INTENTION RECOGNITION THROUGH NATURAL MOTION FOR INTUITIVE HUMAN-ROBOT INTERACTION

A thesis submitted toward the degree of
Master of Science in Engineering

by

Nadav D. Kahanowich

This research was carried out at Tel Aviv University
in the School of Mechanical Engineering
Faculty of Engineering

under the supervision of Dr. Avishai Sintov

March 2022

To my family, for being there.

Abstract

Many tasks performed by two humans require mutual interaction between arms such as handing over tools and objects. In order for a robotic arm to interact with a human in the same way and for other applications of Human-Robot Collaboration (HRC), it must reason about the location of the human arm and hand-held object in real-time. Furthermore and to acquire interaction in a timely manner, the robot must be able predict the final target of the human and detect if and what type of object is held by the human, in order to plan and initiate motion beforehand.

In this research, we explore the use of two low-cost wearable devices. The first device is equipped with an array of Force sensitive resistor (FSR) sensors, to classify hand-held objects. The second is equipped with two inertial measurement units (IMU) for learning reaching motion. The wearable devices can replace or be complementary to visual perception in cases of bad lighting or occlusions in a cluttered environment. For each of the systems, we have collected large data sets from real human motions. For grasped object classification system, we have trained and optimized an ANN and proposed a novel algorithm which increases prediction robustness. For the second system, we have trained and optimized two separate models (ANN and RNN) to estimate the wrist position and predict user reach, with promising results.

Acknowledgments

I wish to personally thank my supervisor, Dr. Avishai Sintov, your vision, creativity and enabling atmosphere were an inspiration to me. The professional and technical guidance were a key element in creating my academic path.

In addition, I would like to acknowledge my family; Yoav, Tali, Tomer and Noa kahanowich, which are the reason I carry on.

and of course, this research would not have been possible without these amazing group of people: My dear friends and lab co-workers, Anton Gurevich, Omer Keinan, Osher Azulay, Itamar Mishani, Noam Nahum, Elran Aronovich, Zohar Danielpur, Alon Mizrahi, Maxim Monastirsky, Eran Bamani, for wise advice and great times.

Our lab engineer, Inbar Ben-David, for the technical support. The amazing staff of the faculty, Andargie 'Molo' mulugeta, Hila Maestro and Oleg Kaganovich. And of course - Kfir Shapira my comrade and partner in crime.

Table of Contents

Nomenclature **iii**

List of Figures **iii**

List of Tables **v**

1: Introduction **1**

1.1 Hand-held object classification 1

1.2 Wrist target prediction 3

2: Related work **5**

2.1 Hand-held object classification 5

2.2 Wrist target prediction 6

3: Hand-held object classification **8**

3.1 System 8

3.1.1 Wearable FMG device 8

3.1.2 Neural-Network classifier 9

3.1.3 Iterative Classification 10

3.2 Experiments and Analysis 14

3.2.1 Model evaluation 14

3.2.2 Sensor placement analysis and Feature Importance 16

3.2.3 Iterative classification analysis 18
3.2.4 Sensor failure 21
3.3 Conclusions 22

4: Wrist location prediction 23

4.1 Algorithm and data flow 24
4.1.1 Arduino script 24
4.1.2 Data acquisition script 24
4.1.3 MLP scripts 24
4.1.4 LSTM scripts 25
4.1.5 Live use script 26
4.2 Learning human motion 27
4.2.1 Problem formulation 27
4.2.2 System 28
4.2.3 Electrical scheme and sensors 28
4.2.4 Mechanical design 29
4.2.5 Data collection and formation 29
4.2.6 Wrist position model 29
4.2.7 Long-Short Term Memory (LSTM) 31
4.2.8 Target prediction Model 32
4.2.9 Curriculum learning 34
4.3 Experiments 36
4.3.1 Wrist position prediction evaluation 38
4.3.2 Target prediction evaluation 39
4.3.3 HRC Demonstration 41
4.4 Conclusions 44

References 45

List of Figures

3.1 Prototype of the FMG wearable device made of two parts for the upper and lower forearm. Each part includes a set of force-sensitive resistors (FSR) designed to sense perturbations of the musculoskeletal system. 9

3.2 Five objects used in the experiments and their typical grasps. From left to right: bottle, mug, screwdriver, scissors and a plate. 14

3.3 (a) Confusion matrix for the Neural-Network classifier (O) with a total success rate of 91.17%. (b) Confusion matrix using IC with classifier O for $\lambda = 0.99$. The total classification success rate is 97.5% with 1.18 average number of iterations. 16

3.4 Classification success rate acquired with regards to the number of episodes recorded. The results show maximum and average values for 10 training attempts while sampling different episodes in each. 17

3.5 Illustration of the sensor locations and importance score computed with the permutation feature importance method. 18

3.6 Confusion matrices for lower success rate classifiers. Classifiers (a) A, (b) B, (c) C and (d) D have a total success rate of 76.88%, 84.77%, 50.10% and 60.12%, respectively. 20

3.7 The classification success rate with regards to the number of iterations for classifiers O, A, B, C and D. 21

4.1 Human user and robot share space $\mathscr{W}$ in which they are to interact or collaborate in a shared task. 27

4.2 System of two IMU positioned on the wrist and upper-arm along with three sets of markers on the shoulder, upper-arm (not used) and wrist, observed by a motion capture system. 30

4.3 (a) A scheme of an LSTM-Pos layer receiving a state sequence $S_H(t) = \{\mathbf{x}(t-H),\ldots,\mathbf{x}(t)\}$ of length H with output vector $\mathbf{h}_j(t)$. The requirement for including the raw data with concatenation (denoted with symbol $\oplus$) would be further analysed. (b) The proposed LSTM-Pos network with stacked layers followed by a feed-forward output layer. The network outputs the predicted target of the wrist $\tilde{\mathbf{p}}_f(t)$ at time t. 33

4.4 Participant 1 collecting data of arm reaching in various torso orientations including facing the collection board, body perpendicular to the board and while sitting down. 36

4.5 Shoulder positions relative to the board (at $y = 0$) before initiating motion (blue) and during motions (red). 37

4.6 Target prediction error (mean and standard deviation) with respect to the number of the past states H. 38

4.7 Typical Heat-map illustrating the mean error (in *mm*) across the collection board when not including wrist position approximation of LSTM-Pos. All models converged their prediction to the center of the board. 40

4.8 Target prediction error (mean and standard deviation) of LSTM-Pos with respect to reaching motion time evaluated on all test episodes. 41

4.9 Heat-maps illustrating the mean error (in *mm*) across the collection board over time and when considering all sensors. Mean error from left to right: $Error(t = 1\ sec) = 116.98\ mm$, $Error(t = 1.33\ sec) = 62.41\ mm$, $Error(t = 1.65\ sec) = 46.86\ mm$ and $Error(t = 2\ sec) = 44.47\ mm$. . . . 41

4.10 Reaching test of Participant 1. 43

4.11 Reaching test of Participant 3. 43

4.12 Motion of participant 2 between two arm poses within the interaction space while the robotic arm follows. 43

4.13 An HRC demonstration in which participant 1 hands-over a bottle to the robot. 43

List of Tables

3.1 Success rate comparison for different classifiers 15

3.2 Classification success rate with regards to different placements of FMG sensors 17

3.3 Importance score for the sensor on the FMG device 18

3.4 Results of IC with different NN classifiers ($\lambda = 0.99$) 20

3.5 Results of IC with different faulty sensors ($\lambda = 0.99$) 21

4.1 Approximation accuracy of wrist position during reaching motion 37

4.2 Target prediction accuracy for LSTM and LSTM-Pos 39

4.3 Arm lengths of all participants 42

4.4 Success rate for reaching experiment 42

1 Introduction

When two humans perform a shared task, each has an ability to predict intentions of his peer with-out verbal communication. Once one human sees the motion of his human fellow, usually his arms and manipulated objects, the intended upcoming task can be predicted for further interaction [50]. For instance, when a human is handing-over an object, his fellow can infer about the reaching target and initiate supporting motion to obtain it. When assembling two parts together, a human assistant can hold one part for support or can handover appropriate tools. Similarly, an upper-limb amputee would need another hand to open a bottle. In Human-Robot Collaboration (HRC), robotic arms should do the same to support a human in completing shared tasks [1]. Having a robot infer about the upper-limbs motion of a human has applications in hand-over activities [66], collaboration in shared workspaces [38], collision avoidance by the robot [33] and virtual reality [19].

A major attempt has been made to integrate HRC approaches in robotics to support humans with disabilities [10] or in performing tasks that require more than one participant [21]. A challenging problem in HRC is to signal the robot of a desired assistive task efficiently and naturally. Some HRC solutions, however, provide non-intuitive control methods such as human gestures [20] or sensing brain activities [44]. Another common approach is Electromyography (EMG) [29] in which electrical signals from the muscles are measured through electrodes and translated to limb movements. In this work, the goal of user intention forecasting for assistive tasks was separated into two sub tasks, handheld object classification and wrist position prediction.

1.1 Hand-held object classification

This part of the thesis is engaged in hand-held object classification. We aim to create a non invasive system where a robotic system can infer about the object held by the human user, while allowing the human user a natural and fluent work flow. We propose a wearable

Force-Myography (FMG) device that is positioned on the human forearm and measures musculoskeletal activities. These activities can imply about the object that is grasped by the human. We use FMG rather than EMG technology due to the latter's requirement of large and costly equipment, and its accuracy can be compromised by sweat, electrode placement and crosstalk [18].

Early work by Amtf et al. [2] have introduced the use of body-worn force sensors to identify patterns in forearm muscle activities. Further work has shown the possibility to identify hand gestures using FMG signals [45], [35]. FMG measures perturbations of the musculoskeletal system and has been reported to be simple to acquire with a relatively high- accuracy [31]. Consequently, acquired FMG data was used in data-based classification of hand gestures [67]. However, data was recollected, and a classifier was trained each time that the sensors have been placed on the arm. Hence, once the sensors have been dislocated, the previously trained classifier significantly loses its accuracy. As opposed to EMG, FMG requires low-cost sensors and a simple portable acquisition device (e.g., Arduino board), and is less sensitive to sensor positioning variations [18], [26]. Furthermore, and according to the author's knowledge, there has not been any attempt to reason about objects within-hand through FMG measurements which is crucial for identification of intended tasks in HRC. As a part of this work, we use FMG measurements to identify objects in a human hand which, in turn, imply about the intended task.

An object is characterized by its geometry and weight which are reflected by the musculoskeletal state of the arm grasping it. Hence, we investigate the ability of a classifier, trained with measured FMG signals, to classify a grasped object from a given set of objects. We aim to rely solely on a low-cost FMG device directly strapped on the human forearm to provide an affordable solution. Previous work included either lower [53] or upper forearm [64] FMG bands. Yet, it is not clear what are the sensing locations required for accurate and robust predictions. Hence, we provide an analysis of the classification accuracy with regards to the placement of the sensors. In addition, we hypothesize that better coverage along the forearm will augment the model and provide better accuracy. Hence, the lightweight wearable FMG device incorporates 15 force-sensitive resistor sensors placed on the lower and upper forearms. Data collected from the FMG device is further used to train a robust classifier to identify objects in hand. We focus on observing object classification using FMG for a single participant while leaving global classifiers for future work. We show that the classifier is robust to re-positioning of the device, i.e., once the classifier has been trained over collected data, it maintains its accuracy even if the device has been taken off previously.

While we show that we can acquire a relatively accurate and robust classifier of objects in hand, we propose an Iterative Classification (IC) algorithm to further improve classification performance. IC is used to increase prediction certainty by sampling additional FMG data. We exploit the continuous time frame in which the user holds a certain object making more samples instantly available. The iterative method can be used with any classifier that provides a class probability distribution. Under some conditions, the iterative process will improve accuracy and robustness even for a classifier that does not provide high success rate by itself. Hence, IC can be exerted on classifiers trained with insufficient data or over a non-optimal classifier model. To conclude, the main contribution is an approach that enables accurate and robust classification of objects in a human hand using affordable, lightweight, and easy to use hardware. The proposed algorithm can provide fast and reliable results in real-time which is essential for practical HRC. For a task planner to decide about a future robot assistive manipulation, it must first be informed of the object in the human hand. The object provides significant information about the upcoming task even before the human arm has begun to move and enables a substantial reduction in the set of possible actions to be performed by the human. Hence, the robot can infer about future actions of the human beforehand and plan a trajectory accordingly [47]. The plan will be updated in real-time with more information about the human and object motion.

1.2 Wrist target prediction

This part of the thesis is engaged in wrist target prediction. We propose an IMU based system which lays on the human user's arm and can predict the location of the wrist, approximately 1.5 seconds in advance. We used an Long Short-Term Memory (LSTM) based model with convolutional layers, trying to optimize the distance between the location predicted by the model and the one measured by a motion camera system. Human motion inference and prediction have been given much attention in the past few decades. Seminal work by Flash and Hogen [16] has suggested that human arms move from point to point in a smooth trajectory while minimizing the mean-square jerk. Others proposed models that include minimum torque [60] and position variances [23]. Nevertheless, these models were demonstrated in a limited and specific setting while human motion is difficult to predict due to the randomness and complexity of human behavior [48]. Therefore, a significant amount of research has been put on the use of Kalman filter variants [30, 69] and data-driven models [62, 36]. In an example for the latter, the work of Landi et al. [32] combined the minimum jerk model with an Artificial Neural-Network (ANN) to predict arm motion based on camera perception. However, all above methods rely on visual perception to acquire human arm pose. Relying on continuous visual feedback limits the

performance of various tasks in which visual uncertainty (e.g., poor lighting or shadows) or occlusion may occur. Moreover, dealing with visual sensing requires a large amount of data and strong computing capabilities [59].

To bypass the challenges of vision, motion prediction using wearable sensors has also been exhibited. For instance, Electro-Myography (EMG) [8] and brain-computer interface [41] have been tested for intention prediction of motor behavior. Wearable Inertial Measurement Units (IMU) have also been proposed and yielded significant results [12]. The work in [58] fused IMU measurements with depth camera (Kinect) perception based on the Unscented Kalman Filter (UKF). Yun and Bachmann [69], on the other hand, filtered two IMU sensors on the upper-arm and forearm to approximate the orientation of the arm. Similarly, Atrsaei et al. [4] used two IMU sensors on the upper-arm and forearm along with UKF to approximate the pose of the arm. However, the methods were focused on approximating the current pose of the arm and did not consider future arm trajectories. In addition, the ability of the above methods to generalize to various participants has not been exhibited.

In this work, we aim to rely solely on a low-cost wearable device directly strapped on the human arm to provide an affordable solution. Achieving vision-free pose estimation and prediction would enable further fusion with vision for better estimation in unstructured or partly-occluded environments. Therefore, we explore the sole use of IMU for learning reaching motion of a human arm. With two IMU sensor located on the upper-arm and forearm, we observe the required data and learning model to predict the target of the reaching arm at early stages of motion in real-time. Furthermore, a learning approach is proposed based on the LTSM model. We investigate the use of raw data from the IMU's along with temporal pose predictions of the arm acquired from a secondary ANN model. Therefore, our proposed method does not require integrating the measured accelerometers to predict the position. Integration is often subject to drift and sensitive to typical noisy signals [58]. Also, and contrary to the common approach, our method does not require the use of filters. We also observe the robustness of the method to taking-off and re-positioning the device on the arm, and for several users.

The main contribution is an approach that enables real-time target prediction in reaching motions of a human arm using affordable, lightweight, and easy to use hardware, and without the need for visual perception. With early real-time information, the robot could promptly initiate a response motion to interact in a timely manner. While we focus on collaboration with robotic arms, the approach can also be applicable to interactions with prosthetic hands, drones, teleoperation, and virtual reality.

2 Related work

In HRC, common control methods that have been used to efficiently signal the robot of a desired assistive task are quite limited. Many assistive robots are specially designed for a specific task in a priori known environment [40]. Other assistive arms use non-intuitive control methods such as predefined human gestures and gazes [20, 43] or brain-computer interface that sense brain activities [44]. Specially designed gloves such as in [14] measure both acceleration and flexion for motion capturing and virtual reality. However, these gloves limit the tactile sensation of the user and thus, not suitable for general use. A widely researched approach is to acquire and classify neurological activities through Electro-Myography (EMG) [29, 8, 57]. EMG detects electrical signals generated by muscle tissue and implies the human subject's intention. Even though this method frees the hand and allows full tactile sensation, it usually requires expensive and highly sizable equipment [18]. In addition, different artifacts and cross-talk may decrease the quality of the signal [3].

The field of HRC contains a large variety of sub-fields. In this work, we tackle the task of object handover and human intention recognition. Leaning on HRC developments and trends [9, 34].

2.1 Hand-held object classification

Inferring about the object being used by the user is crucial for HRC tasks, once the robotic system can identify the handheld object, it is possible to confine the possible intentions of the user to a finite number of actions. As mentioned, there are several possible techniques to approach this task. In this work, we utilize Force-Myography (FMG), where force sensors capture radially directed force distributions through expansion and contraction of the musculoskeletal system [11]. Prior works [2, 45] have shown that exterior sensing of muscle surface perturbations incorporates important information about task activity. A work

study in [35, 31] has advanced the idea by identifying finger motions from muscle perturbation via force sensors on the forearm. While their methodologies did not allow having a wearable system for real-time feedback, these studies have established the feasibility of using FMG for monitoring upper-extremities gestures. More recently a wearable device has been proposed [64] that is composed of a linear set of eight force sensors. A classifier was trained to identify in real-time hand postures. Similarly, a wearable FMG feedback system was used to detect four basic hand motions in rehabilitation analysis [67].

2.2 Wrist target prediction

Motion prediction based on extracted features of human motion can be divided into two categories, model-based and model-free. Model-based approaches establish an analytical function for mapping between measured features to the kinematic or dynamic behavior [17]. Model parameters are studied and refined through experiments but are challenging to evaluate accurately. On the other hand, the model-free approach is a black-box mapping acquired through machine learning [39]. A large amount of data is used to train an Artificial Neural-Network to map sensed features to the current gesture or posture of the hand. As such, the works in [52], [63] trained a neural network to map EMG signals to joint angles of the upper limb. A major part of prior work focused on the use of visual perception to investigate human arm motion. As such, Oguz et al. [46] solved an inverse optimal control problem to derive the true cost function that governs a set of motions. The work in [56] predicted to which object the human hand is reaching. The target object was predicted by observing arm motion with a camera in a discretized workspace. It is also worth mentioning extensive attempts for pose and motion prediction of the entire human body through visual perception [17, 39, 52]. Similarly, depth camera such as Kinect is an alternative solution using spatial point cloud [63].

While the above focused on extracting mathematical models of motion, a different approach involves data-driven models. The work of Cheng et al. [36] proposed the use of semi-adaptable ANN to learn a human arm transition model and adapt it to time-varying human behaviors. Another approach uses Hidden Markov Models to approximate human pose or occupied workspace based on visual observations [62, 15]. The use of Recurrent Neural-Networks (RNN) is also a common approach where sequential temporal data is used to predict motion trajectories [37, 70]. Liu and Liu [36] combined a Modified Kalman Filter to adapt an RNN model to changes in environmental conditions. The work in [49] used a database of recorded human motions to predict in real-time intended targets in reaching motions. However, and as previously indicated, relying solely on continuous

visual feedback limits the performance in unstructured environments where occlusions may occur. Therefore, in this work we focus on observing human arm motion through wearable sensors while later fusion with visual perception may provide a complete solution. Object hand-over from a robot to a human, and vice versa, is a potential application which requires reasoning about human arm pose and future trajectory. Visual perception is the leading method to locate the human arm and approximate its pose [51]. In [66], human-to-robot handovers were conducted by visually classifying a human grasp of the reached-out object and planning an approach trajectory for the robot. Nemlekar et al. [42] approximated the object transfer point based on observed human behavior and motion.

3 Hand-held object classification

This chapter discusses the first sub-task of hand-held object classification. It covers the system design (mechanical, electrical and scripts), data flow, results and conclusions. **The work in this chapter was published in the IEEE Robotics & Automation Letters [27] and was presented (virtually) at the IEEE Conference on Robotics and Automation, 2021.**

3.1 System

3.1.1 Wearable FMG device

As previously described, prior work included either upper or lower forearm bands. To improve accuracy and to analyze the dominant measurement locations, we have combined both and fabricated a wearable device with wider coverage along the forearm). The device is composed of 15 low-cost Force-Sensitive Resistors (FSR), model FSR-402 by Interlink Electronics. FSR sensors are made of polymer films that vary their electrical resistance upon changing pressure on their surface. The device consists of three main components: (a) upper forearm band with six FSR sensors, (b) lower forearm band with nine sensors organized in two rows and (c) a data acquisition system based on an Arduino Mega 2560 board. The FSR sensors were positioned in equal spacing along the bands as seen in Figure 3.1. We note that the device includes two bands for prototyping considerations but can easily be fabricated as one unit. The bands were fabricated by 3D printing with an elastic polymer (Thermoplastic elastomer). They include a flexible bulge for each sensor to ensure proper attachment to the skin while maintaining flexibility during arm motion. Each FSR sensor is connected to an analog pin of the Arduino through a voltage divider of 4.7kΩ resistor. Such acquisition configuration provides real-time data stream of all the given sensors in a frequency of up to 300Hz. The described system is composed of low-cost and light-weight hardware which is appealing and suited for easy arm movements.

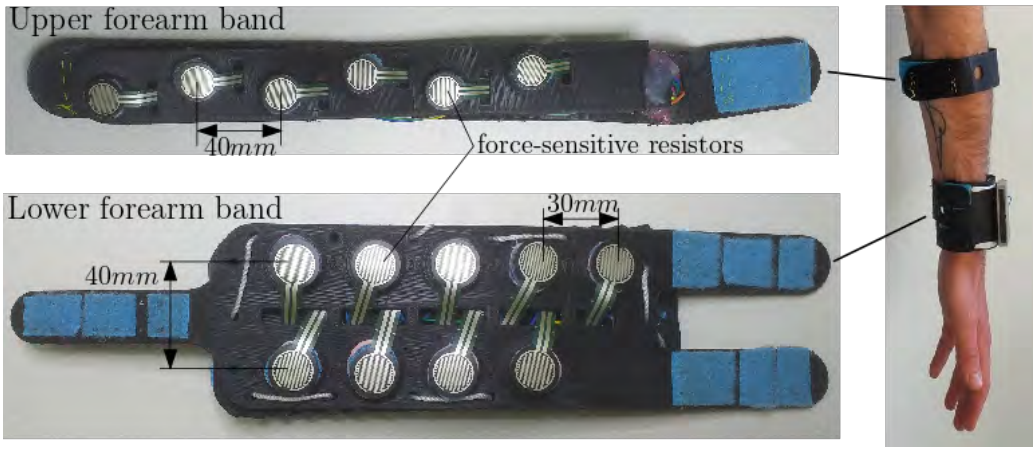

Figure 3.1: Prototype of the FMG wearable device made of two parts for the upper and lower forearm. Each part includes a set of force-sensitive resistors (FSR) designed to sense perturbations of the musculoskeletal system.

3.1.2 Neural-Network classifier

We aim to identify an object grasped by the human solely by measuring FMG signals measured by the device. Given a set of m objects $\{\mathscr{O}_1, \ldots \mathscr{O}_m\}$, we require to identify an object from the set. That is, we require real-time classification based on pattern recognition of the input signals. This is achieved through supervised learning with the use of a feed-forward Neural-Network (NN) trained over labeled signals.

Let $\phi \in \mathbb{R}^n$ be the observable state of the musculoskeletal system measured by the FMG system with n FSR sensors. For each object $\mathscr{O}_i$, training data is collected by holding it as seen in Figure 3.2. We record grasps of objects as intended during tasks, e.g., grasping scissors by their ring handles. To increase data variance, the data was recorded in various arm postures, i.e., while arbitrary moving the shoulder, elbow and wrist. Ultimately, the resulting training data is a set of M labeled FMG signals $\Phi = \{(\phi_1, l_1), \ldots, (\phi_M, l_M)\}$ where label l_i corresponds to object $\mathscr{O}_{l_i}$. However and as described previously, we aim to acquire a model robust to replacing of the FMG device. Recording data after a one time positioning of the device will not be robust and a trained model is most likely to fail after taking-off and re-positioning.

Classification failure while using the above formulation will happen for two reasons: inability to re-position the device at the same location and inability to tighten it with the same forces each time. For the former challenge, we collect data in N episodes where, in each episode, the device is taken-off and re-positioned. To cope with the different tightening forces at each episode, we consider episode values relative to the initial forces after strapping in. Thus, at the beginning of episode j, the user is required to perform a simple

calibration process in which the muscles are relaxed prior to picking up the object. The FMG baseline signal $\phi_o^{(j)}$ during the relaxation is subtracted from the episode measurements to decrease variance, i.e., a signal is now given by $\tilde{\phi}_i^{(j)} = \phi_i^{(j)} - \phi_o^{(j)}$. Next, since the data is significantly noisy, we apply a simple Mean Filter of width w to each sensor measurement. A Median filter exerted similar results. However, while Mean filters suppress noise, unique features of the data may be lost. Hence, a multitude of measurement vectors of different classes may be described with the same mean, leading to low classification accuracy. Therefore, we include the standard deviation of each sensor measurement which retrieves some previously lost distinctive features of the class. Hence, a processed signal of episode j is now $\mathbf{x}_i^{(j)} \in \mathbb{R}^{2n}$ including both mean and standard deviation of n signals along a window of length w.

The labeled dataset is now composed of N episodes $\Phi = \Phi_1 \cup \Phi_2 \cup \ldots \cup \Phi_N$ where $\Phi_j = \{(\mathbf{x}_1^{(j)}, l_1), \ldots, (\mathbf{x}_M^{(j)}, l_M)\}$. Dataset Φ can now be used to train a NN mapping to classify a set of objects, i.e., train map $h : \mathbb{R}^{2n} \rightarrow [0,1]^m$ such that a class probability distribution $\mathbf{p} = \{p_1, \ldots p_m\}$ (where $\sum_m p_i = 1$) using $h(\mathbf{x})$ is computed by $\mathbf{p} = h(\mathbf{x})$. Next, we present an iterative classification method in which models, even with low certainty, can be improved by sequentially drawing more samples.

3.1.3 Iterative Classification

A trained classifier may yield object identification with low certainty due to noisy measurements, insufficient measurement data or non-optimal model. Nevertheless, we may exploit the continuous time frame in which the user holds a certain object. While common classification tasks rely only on one sample for prediction, in our case, we may rapidly acquire additional samples while being certain that they originate from the same class. Consider FMG signals arriving sequentially $\{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$ in real-time while holding an unknown object. It is required to estimate conditional probability for class $\mathscr{O}_i$ given k sequential samples, i.e., $P_k(\mathscr{O}_i | \mathbf{x}_1, \ldots \mathbf{x}_k)$. We propose to use an iterative process where a score is given in each iteration based on prediction certainty. Unlike sequential Bayesian update [28], the proposed approach can be used with any type of classifier that provides a probability distribution.

The *Iterative Classification* (IC) process is described in Algorithm 1. We track the scores of the classes based on the predictions for each sample provided by any chosen classifier. We maintain a vector $\mathbf{s} = (s_1, ..., s_m)$ of cumulative scores for the classes. In each iteration, a signal $\mathbf{x}$ is sampled, followed by acquiring a class probability distribution $\mathbf{p} = h(\mathbf{x})$. Generally, function h can be any trained classifier that outputs a probability

distribution of the prediction. The probability output is considered as the certainty of the classifier to its prediction and the score to the class prediction. Hence, the highest probability p_i in $\mathbf{p}$ is the iteration score for class i and is added to s_i. This process is repeated until the normalized cumulative score $\hat{s}_{max}$ for some class reaches above a lower bound $\lambda \in [0,1]$. It is also possible to *fully accumulate* (FA) all class scores by updating $\mathbf{s}$ with all iteration probabilities, i.e., replace lines 5-6 in Algorithm 1 with $\mathbf{s} \leftarrow \mathbf{s} + \mathbf{p}$. However, this will result in an excessive number of iterations for ill-trained classifiers while requiring a carefully tuned λ. For a sufficiently trained classifier, FA would provide only marginal accuracy improvement as will be seen in the experimental section.

Algorithm 1: `iterative_classification`(λ)

1 Initiate elements of $\mathbf{s} = \{s_1, ..., s_m\}$ to 0;
2 **repeat**
3 $\quad \mathbf{x} \leftarrow$ `sample`();
4 $\quad \mathbf{p} \leftarrow h(\mathbf{x})$;
5 $\quad i \leftarrow arg\max(\mathbf{p})$;
6 $\quad s_i \leftarrow s_i + p_i$;
7 $\quad o \leftarrow arg\max(\mathbf{s})$;
8 $\quad$ **if** *first iteration* **then**
9 $\quad\quad \hat{s}_{max} \leftarrow s_o$;
10 $\quad$ **else**
11 $\quad\quad \hat{s}_{max} \leftarrow s_o / (\sum_i s_i)$;
12 **until** $\hat{s}_{max} > \lambda$;
13 **return** o ; `/* return class index */`

Let $P(l = j|\mathscr{O}_i)$ be the probability for classifier h to assign label j to a grasped object $\mathscr{O}_i$ such that

$$\sum_{j=1}^{m} P(l = j|\mathscr{O}_i) = 1. \tag{3.1}$$

A sufficiently trained classifier must satisfy

$$P(l = i|\mathscr{O}_i) > P(l = j|\mathscr{O}_i) \tag{3.2}$$

for any $i, j \in \{1, ..., m\}$ and $j \neq i$. Naturally, higher values of $P(l = i|\mathscr{O}_i)$ for all $i \in \{1, \ldots, m\}$ mean a more accurate classifier. In many cases, prediction probability of incorrect predictions tend to be lower than the prediction probability for correct examples [24]. Hence, given $p_{max} = \max(\mathbf{p})$, the expected value for p_{max} when successfully classi-

fying object $\mathscr{O}_i$ would be larger than an erroneous prediction. That is, statement

$$\mathbb{E}(p_{max}|l=i,\mathscr{O}_i) > \mathbb{E}(p_{max}|l=j,\mathscr{O}_i) \tag{3.3}$$

holds for any $j \neq i$. According to Algorithm 1, vector $\mathbf{s}$ accumulates scores for class predictions with the increase of iterations. In addition, a score is given to s_j only if label $l = j$ is assigned to the query object in a particular iteration. Hence, the expected normalized value $\hat{s}_j$ of component s_j given object $\mathscr{O}_i$ is

$$\mathbb{E}(\hat{s}_j|\mathscr{O}_i) = \mathbb{E}(p_{max}|l=j,\mathscr{O}_i)P(l=j|\mathscr{O}_i) \tag{3.4}$$

for any $j \in \{1,\ldots,m\}$ and where $\hat{s}_j = s_j/(\sum_i s_i)$. One may view $\hat{s}_j$ as the probability approximation for grasping $\mathscr{O}_j$ after some number of iterations, i.e., $\hat{s}_j \approx P_k(\mathscr{O}_j|\mathbf{x}_1,\ldots\mathbf{x}_k)$. From (3.2)-(3.4), it must be that

$$\mathbb{E}(\hat{s}_i|\mathscr{O}_i) > \mathbb{E}(\hat{s}_j|\mathscr{O}_i), \;\; \forall j \neq i. \tag{3.5}$$

While we acknowledge that class probability distributions outputted from a *softmax* layer may not always reflect the true certainty of its prediction [22], preliminary results show that condition (3.3) holds in our case. Nevertheless, even with a more strict assumption, where p_{max} for any prediction (erroneous or not) has a uniform distribution $p_{max} \sim U(\frac{1}{m},1)$ such that

$$\mathbb{E}(p_{max}|l=i,\mathscr{O}_i) = \mathbb{E}(p_{max}|l=j,\mathscr{O}_i) = \frac{m+2}{2m}, \tag{3.6}$$

statement (3.5) remains valid due to (3.2).

The above statements imply that as long as a classifier satisfies (3.2), the expected cumulative score $\mathbb{E}(\hat{s}_i|\mathscr{O}_i)$ will increase and converge to an higher value than $\mathbb{E}(\hat{s}_j|\mathscr{O}_i)$ ($j \neq i$), with the increase of classification iterations, i.e., $\hat{s}_i$ is more likely to be equal to $\hat{s}_{max}$. Hence, the certainty about the prediction will grow with the addition of more samples. In turn, this will result in continuous improvement of the classifiers success rate. Let $m_p \in \{0,1,\ldots,m\}$ be the number of grasped objects that satisfy (3.2) for a given classifier. If a classifier is sufficiently trained such that $m_p = m$, condition (3.2) will be satisfied for all objects. Hence, the total success rate would converge to 100% with the increase of iterations. However, when $m_p < m$, condition (3.2) is satisfied only for the m_p objects. Consequently, the classification success rate will increase only for these objects while declining for the remaining $m - m_p$ ones. This means that the success rate upper

limit for a certain classifier is

$$\xi = \frac{m_p}{m} \times 100\%. \tag{3.7}$$

The convergence rate depends on the quality of the classifier, that is, on the accuracy $P(l = i|\mathscr{O}_i)$ and prediction certainty $\mathbb{E}(p_{max}|l = i, \mathscr{O}_i)$, for all i.

The proper amount of iterations to reach some level of accuracy or certainty is not known beforehand. Hence, when required to acquire a classification in a short period of time, we cannot set the termination criterion to some arbitrary number of iterations. Therefore, Algorithm 1 sets a termination criterion when reaching some certainty above a threshold λ. Alternatively, classification can be done without a termination criterion for a long time horizon with continuous certainty improvement. The collaborative robot, however, will need to identify task completion by other means. These will be shown and analyzed through experiments in the next section.

3.2 Experiments and Analysis

In this section, we test and analyse the proposed FMG device and classification method over a set of objects. We have picked five everyday objects shown in Figure 3.2 including a bottle, a mug, a screwdriver, scissors and a plate.

Figure 3.2: Five objects used in the experiments and their typical grasps. From left to right: bottle, mug, screwdriver, scissors and a plate.

A training set is acquired by recording $N = 40$ episodes for each object with a single participant. The participant grasped the objects in a task-based pose (e.g., by the handle of the screwdriver) similar to the taxonomy described in [13]. In addition, the FMG device is taken off and re-positioned between the episodes. While doing so, the object is also put down and picked up again in a task-based grasp with some variations. It is important to note that the data is recorded while the arm is moving through various configurations to increase variance and include different grasping postures. For each episode and object, a batch of M =10,000 data points is recorded and labeled. All data is pre-processed with $w = 100$, resulting in dimension 30 (combining mean and standard deviation of an $n = 15$ dimensional measurement). 10% of the data was dedicated for testing and hyper-parameters optimization, and not included in the training. Similarly, we recorded and pre-processed five episodes for each object to form a validation set yielding 4,950 samples for each object and 24,750 in total. We use the validation data to test classification success rate in a standard fashion over all tests. The validation data was completely decoupled from the training and testing phases. While we perform validation tests off-line, data is taken sequentially as recorded as if done in real-time.

3.2.1 Model evaluation

Using the training data, we have optimized a feed-forward NN classifier. The resulting NN has two hidden layers of 398 neurons each and a Rectified Linear Unit (ReLU) activation. A dropout of 50% and an L2 regularizer (factor 10^{-5}) were included to control

Table 3.1: Success rate comparison for different classifiers

	Classification success rate	
Classifier	w/o pre-process	w/ pre-process
Nearest Neighbors	75.19%	83.68%
Naive Bayes	64.41%	62.11%
Linear SVM	75.52%	80.86%
Random Forests	70.90%	63.29%
Neural-Network	63.01%	91.17%
Decision Trees	60.95%	63.59%
AdaBoost	72.38%	73.39%
LDA	73.68%	78.99%

apparent overfitting. Additionally, the ADAM optimizer was used with a sparse categorical cross-entropy loss function. The network was trained with the back-propagation algorithm. Furthermore, we have conducted a comparison to other common classifiers, including: Nearest-Neighbors, Naive Bayes, Support Vector Machines (SVM) with a linear kernel, Random Forests, Decision Trees, Adaptive Boosting (AdaBoost) and Linear Discriminant Analysis (LDA) [54].

Table 3.1 reports the classification success rate for these classifiers over the validation data. We note that these are the rates for one-time calls without IC, i.e., classification according to one sample as input. The table also shows the importance of the pre-processing step (no Mean filter and standard deviation inclusion). The pre-processing step slightly improves in most cases with a significant increase for the NN classifier. All tests, however, include signal substraction by the initial episode relaxation measurements to compensate varying tightening forces. Without filtering the training data, the most accurate NN with input dimension of 15 reach success rate of 63.01%. When applying the Mean filter, the success rate significantly increases to 88%; and 91.17% when also including standard deviation. The results indicate that both mean and standard deviation values embed valuable data, where the mean values are of greater importance for the NN performance. Overall, it is clear that the NN classifier outperforms and, therefore, further used in our experiments.

Figure 3.3a presents the confusion matrix for the NN classifier, denoted as classifier O, with a total success rate of 91.17%. The most erroneous classification is for the scissors in which some grasp variations can be confused with a plate or a screwdriver. This motivates the observation of multiple signals through time to increase certainty and success rate.

In the next analysis, we wish to observe the robustness property with regards to the number of recorded episodes. Recall that each episode contains 10,000 recorded signals and the FMG device is removed and re-positioned before each episode. In order to observe

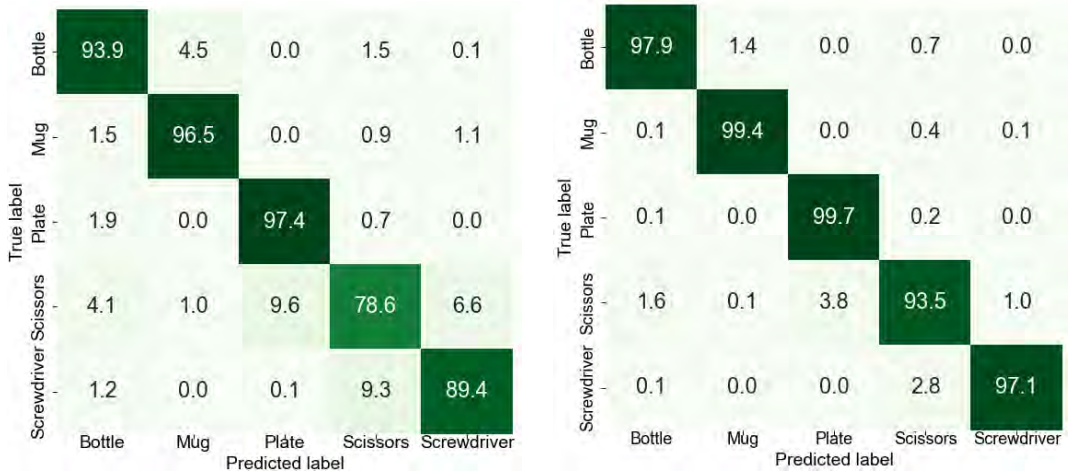

Figure 3.3: (a) Confusion matrix for the Neural-Network classifier (O) with a total success rate of 91.17%. (b) Confusion matrix using IC with classifier O for $\lambda = 0.99$. The total classification success rate is 97.5% with 1.18 average number of iterations.

the success rate with regards to the number of episodes N, the NN classifier was repeatedly trained over 10 trials for each given $N = 1, ..., 40$ while sampling different episodes from the dataset in each trial. Results can be seen in Figure 3.4. The classification success rate improves with the increase of episodes, reaching approximately 90% success rate with more than 30 episodes. Also, the standard deviation over the 10 trials decreases as the number of episodes rise. This behaviour confirms the main issue dealing with data originating from body-muscle related source; the placement and replacement after every episode creates considerable variations to the data. As expected, additional episodes in the training data results in a more robust classifier to variations in device placements and tightening forces.

3.2.2 Sensor placement analysis and Feature Importance

As discussed in the introduction chapter, previous work have positioned FSR sensors either on the lower or upper forearm. We now investigate the contribution of each sensor location to the success rate of the classification. Hence, we compare between various configurations of FMG measurements: an upper forearm band (UF), a lower forearm band with one row of sensors (LF-1), a lower forearm band with two rows of sensors (LF-2) and using all sensors in the FMG device. For each configuration, we optimized the NN hyper-parameters. Classification success rate is summarized in Table 3.2. We see that sen-

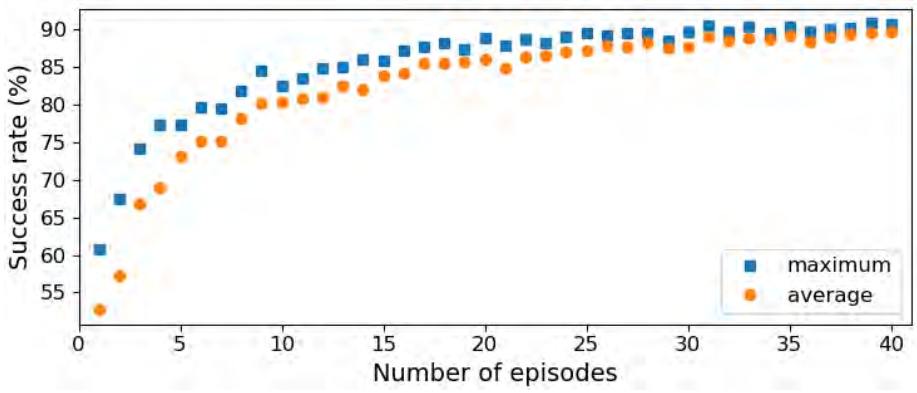

Figure 3.4: Classification success rate acquired with regards to the number of episodes recorded. The results show maximum and average values for 10 training attempts while sampling different episodes in each.

sors on the lower forearm contribute the most valuable information while sensors on the upper forearm by themselves are not sufficient.

Table 3.2: Classification success rate with regards to different placements of FMG sensors

	UF	LF-1	LF-2	All (O)
Num. of sensors	6	5	9	15
Success rate	57.20%	75.13%	82.80%	91.17%

To achieve better understanding of the FSR sensors effect on the model's performance, we observe the impact of each sensor on predictions. Permutation feature importance is a common method to quantify the contribution of each feature in an NN [65]. This is done by measuring the increase in the prediction error after permuting the values of each single feature separately, which breaks the relationship between the feature and the true label. We use classifier O and randomly permute the validation data, a single feature each time. The importance score of feature i is the error e_i relative to the non-permuted model. In other words, the score is defined as the decline in accuracy resulting from the permutation of a sensor's values. The score is computed according to $e_i = \frac{q-q_i}{q} \times 100\%$,

where q is the success rate of the non-permuted model and q_i is the success rate when feature i is permuted. The results of feature importance evaluation over 30 repetitions are shown in Table 3.3 and illustrated in Figure 3.5 along with sensor locations. The relative accuracies indicate a relatively strong dependence on the lower forearm sensors and correlate to the results in Table 3.2. The above feature importance correlates with the layout of the forearm muscles. The most significant sensors, 8 and 14, lay on top of the flexor carpi radialis and flexor digitorum superficialis, which have an important role

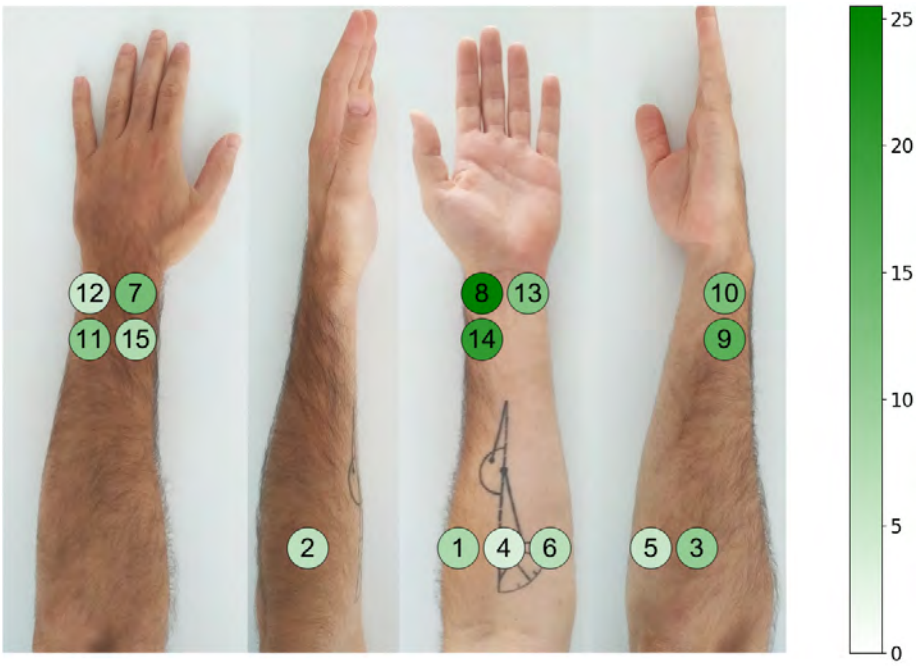

Figure 3.5: Illustration of the sensor locations and importance score computed with the permutation feature importance method.

Table 3.3: Importance score for the sensor on the FMG device

Sensor index	Importance score	Sensor index	Importance score	Sensor index	Importance score
1	8.23%	6	6.90%	11	11.47%
2	6.58%	7	13.62%	12	5.33%
3	10.67%	8	25.50%	13	12.21%
4	3.78%	9	16.50%	14	20.46%
5	5.52%	10	13.16%	15	7.89%

in operating the wrist and fingers. While some sensors are more important than others, the results show that all sensors along the forearm contribute to an accurate and robust classification.

3.2.3 Iterative classification analysis

The results of Section 3.2.1 show the ability to train a classifier with relatively high success rate. However, it may be possible that a model is not accurate enough due to insufficient data or non-optimal NN hyper-parameters. Hence, we now analyse the proposed IC algorithm described in Section 3.1.3 (Algorithm 1) aimed to raise the certainty of the output given by a classifier. We first apply the algorithm to classifier O. When exerting IC with $\lambda = 0.99$, the total success rate increases to 97.5% with 1.18 average number of iterations.

Figure 3.3b shows the confusion matrix of the iterated classifier. Adding more samples increases certainty and, therefore, success rate. Specifically, significant improvement is seen for the scissors and screwdriver when comparing to Figure 3.3a. The above results show that IC along with a sufficiently good classifier can exhibit accurate and robust classifications even after the re-positioning of the FMG device.

We now explore the use of the iterative algorithm for trained classifiers that initially achieved lower success rate. Four classifiers were chosen and their confusion matrices are seen in Figure 3.6. Classifiers A and B were trained with only 4 and 16 episodes, and reached success rate of 76.88% and 84.77%, respectively (confusion matrices are seen in Figures 3.6a and 3.6b). Another two classifiers, C and D (Figures 3.6c and 3.6d), that use the entire training set (40 episodes) but with non-optimal hyper-parameters, reach success rates of 50.10% and 60.12%, respectively. We note that the confusion matrices are an approximation of $P(l = j|\mathscr{O}_i)$ for $i, j \in \{1,2,3,4,5\}$ where the diagonal elements approximate $P(l = i|\mathscr{O}_i)$. Hence, classifiers C and D do not satisfy (3.2).

Table 3.4 presents the success rate improvement when applying IC with $\lambda = 0.99$ over the validation data. The results for classifiers A and B clearly show that a classifier can be trained with less episodes (i.e., smaller sized training set) and, achieve high and robust accuracy with a low number of iterations. The success rates for classifiers C and D show limited improvement while the number of iterations is relatively high. This is the result of the low initial accuracy of the classifiers and inability to satisfy condition (3.2). With FA, the success rate is 98.1% with 1.16 average number of iterations for classifier O, providing only marginal improvement. On the other hand, FA for classifiers A-D failed to converge with a high number of iterations for many trials making the approach infeasible.

We next analyze the performance of the algorithm over a long horizon when removing the λ termination criterion. Figure 3.7 shows the success rate behavior with respect to the number of iterations. Classifiers O, A and B (represented by Figures 3.3a, 3.6a and 3.6b) show satisfaction of condition (3.2). Consequently, Figure 3.7 shows continuous success rate increase toward converging to a common upper limit of $\xi = 100\%$ as expected from (3.7). Classifier O converged to 100% relatively fast while the other two require more iterations due to their lower success rate for some objects. Nevertheless, the algorithm contributes significant improvement compared to the initial results.

As can be seen in Figures 3.6c and 3.6d, classifiers C and D share a common property where they both have exactly one object that does not satisfy condition (3.2), i.e., $m_p = 4$. Therefore and according to (3.7), they are both expected to exhibit convergence to an upper limit of $\xi = 80\%$ success rate. Figure 3.7 indeed shows moderate increase in success rate

Table 3.4: Results of IC with different NN classifiers ($\lambda = 0.99$)

Classifier	*A*	*B*	*C*	*D*	*O*
Initial success rate (%)	76.88	84.77	50.10	60.12	91.17
Success rate w/ IC (%)	87.15	92.39	58.71	69.13	97.5
Difference (%)	10.27	7.62	8.61	9.01	6.33
Avg. iterations	1.31	1.17	2.06	2.04	1.18

for the two classifiers toward 80%. Additional trials show that they reach a success rate of approximately 76% after 1,000 iterations slowly approaching their upper bound.

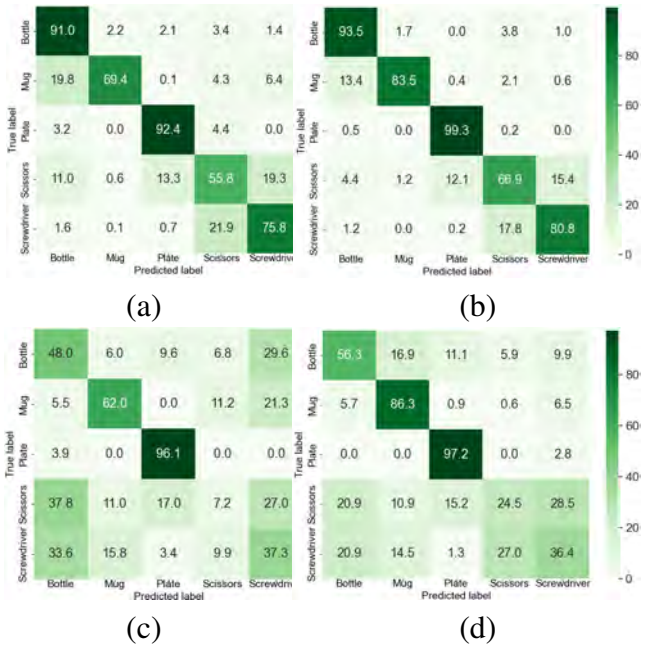

Figure 3.6: Confusion matrices for lower success rate classifiers. Classifiers (a) *A*, (b) *B*, (c) *C* and (d) *D* have a total success rate of 76.88%, 84.77%, 50.10% and 60.12%, respectively.

Finally, we report the computational time required to classify an object within hand. We have experimented on an Intel-Core i7-9700 Ubuntu machine with 16GB of RAM. The average computation time over 5,000 trials when $\lambda = 0.99$ is 24 milliseconds. Alternatively, if the λ termination criterion is not used and the algorithm keeps improving the prediction certainty, iteration frequency reaches 53.5Hz. These results show that accurate classification can be done very fast and in real-time.

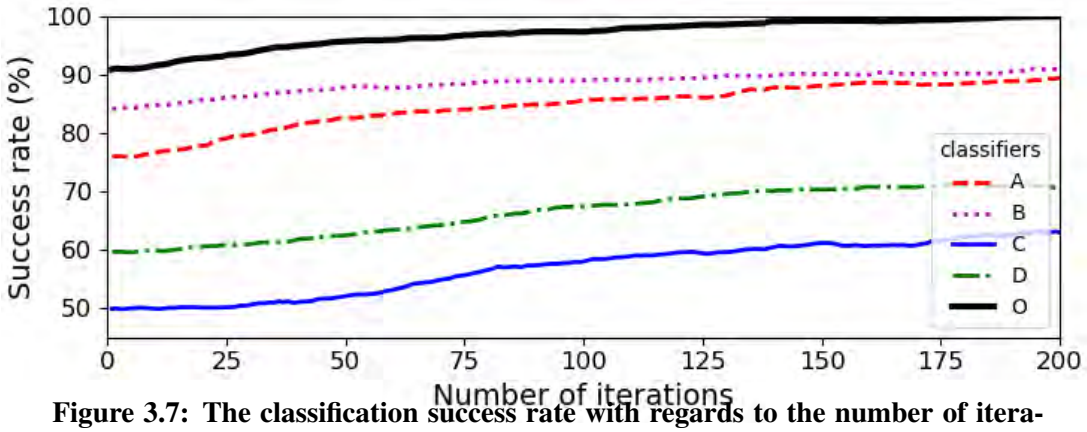

Figure 3.7: The classification success rate with regards to the number of iterations for classifiers *O*, *A*, *B*, *C* and *D*.

3.2.4 Sensor failure

When a sensor in the FMG device fails, a well trained classifier can lose its accuracy. We now test the use of IC with an already trained classifier when various sensors fail. We virtually fail a sensor by setting its values in the validation set to zero. We note that failure can occur with other non-zero values or increased noise, leading to different success rates. Table 3.5 shows the results of several different sensors that fail individually with and without using IC. Indices of the sensors can be viewed in Figure 3.5. We note that these results do not match the general failure importance values (Table 3.3) as they only reflect a special case of setting the failed value to zero. Failure of some sensors lead to significant loss of accuracy. The loss magnitude depends on the impact of the sensor on the classification success rate for specific objects and on the respective feature importance. IC, in such case, can provide valuable improvement while not enough in some cases.

Table 3.5: Results of IC with different faulty sensors ($\lambda = 0.99$)

Sensor index	3	4	8	9	14
Initial success rate (%)	58.17	86.23	78.37	78.11	62.77
Success rate w/ IC (%)	63.22	93.80	89.72	88.03	70.64
Difference (%)	5.05	7.57	11.35	9.92	7.87
Avg. iterations	1.2	1.17	1.22	1.22	1.25

3.3 Conclusions

We have presented the problem of classifying an object in hand using simple FMG measurements accurately and robustly. We have shown the use of 15 FSR sensors on a low-cost FMG device worn on the forearm. Measurements from the sensors are recorded during the handling of various objects while removing and replacing the device. The data is used to train a NN classifier that exhibited high classification success rate. We further proposed an iterative classification algorithm to augment the classification. The IC has shown to significantly improve certainty of predictions by sampling additional signals. Furthermore, success rate is improved even for less accurate classifiers. Hence, a single user can train a robust and relatively accurate classifier quickly. In addition, we have performed an analysis to understand the key locations for FSR sensors. Our proposed method has shown a robot's ability to reason about the object in hand without verbal communication or visual feedback. The FMG device provides fast and robust classification of grasped objects that imply about the intended task of the human. The information about the object will be part of the decision making for a task planner to take assistive actions.

Future work could focus on increasing the resolution of the FSR sensors and reasoning about the pose and weight of an object in hand. In addition, more sensors can be added along the arm to acquire additional information about its pose and to predict future trajectory. We note that some objects with similar geometry and a task-based grasp (e.g., a screwdriver and a spatula) cannot be distinguished by solely observing finger poses. In such case, future work may consider the observation of FMG measurements during the task over time or add context information. A global classifier or model transfer to a new user are interesting topics to further be explored.

4 Wrist location prediction

This chapter elaborates about the second sub-task of wrist location prediction. It covers the system design (mechanical, electrical and scripts), data flow, results, HRC demonstration and conclusions. **The work in this chapter was submitted to the Mechatronics journal.**

4.1 Algorithm and data flow

4.1.1 Arduino script

This script, written in CPP and embedded on the Arduino, control the raw data acquisition from the controller. It uses a list with specific headers for every feature of every IMU. The script uses the multiplexer to switch between the IMUs.

4.1.2 Data acquisition script

This python script oversees receiving the raw data from the Arduino (IMU data, model's input) and the motion capture system data (marker data, labels) The script takes the raw data, sent by the controller, and converts it from byte type into list of floats. It uses the headers to file the features into their designated cells in the input arrays. The script also instruct the user where to reach upon the recording matrix. Once each trajectory recording is done, the script saves it into a pickle file, with a unique file name. The script can identify connection issues to each system (Arduino and motion capture cameras) and reconnect automatically. After this stage, the data is being sent to the scripts which collect the raw recording pickle files and compile then into datasets.

4.1.3 MLP scripts

As mentioned, the raw recording pickle files are being collected by the `Dataset_maker_for_MLP` script. The script has option to include or exclude every one of the 18 IMUs features for fitting the models for the best input feature combination. In addition, the same can be done with the marker input data, which contains position and rotation 3D data for the elbow, shoulder, and wrist. The script contains several pre-processing methods, all can be turned on and off:

- Add noise – this method adds Gaussian noise to the samples, to increase the data's variance.
- Shuffle data – this method shuffle the data from it's original order, this is to prevent certain batches of data, sent to the model, to be samples from a specific area of the data space, resulting with unwanted learning behavior in the trained models.
- Scale – act only upon the IMU data, scales the features between 0 and 1.

- Normalize – act only upon the IMU data, scales the features to have a normal value of 0 with STDev of 1.

- Remove outliers – act on both IMU and marker data, this method uses z score test to identify outliers and remove them from the dataset.

- PCA – act only on the IMU data, is used to reduce the amount of input features, while keeping the essence of the data.

- Marker relative to matrix – this method, acts only on the marker data, moves the origin to the matrix's top middle point.

After using the chosen methods, the script inserts the data into a Data-loader object, for training and testing purposes. In addition, the scaling data, normalization scaler data, PCA and z score data are being saved for future use. Then, the data is loaded by a plotting script, which plot's the various features, both IMU and markers, for visual inspection. The next script in order is the model optimization script, it takes the data-loader files and create studies which try to optimize the model's test loss score. A study generates 1000 trials, each trial is a combination set of hyper-parameters to optimize. The script plots several parameters and scores of each model, train and test loss as a function of epoch, error heat-maps for specific recording matrix cell and specific time step, mean error as a function of time step, predicted trajectory vs. true trajectory 3D graph.

4.1.4 LSTM scripts

Similarly to the MLP work flow, the raw recording pickle files are being collected by the `Dataset_maker_for_LSTM` script. The script has option to include or exclude every one of the 18 IMUs and MLP output features for fitting the models for the best input feature combination. In addition, the same can be done with the marker input data, which contains position and rotation 3D data for the elbow, shoulder, and wrist. The script contains several pre-processing methods, all can be turned on and off, in addition to the methods mentioned in the `Dataset_maker_for_MLP` script, these methods are utilized:

- Moving time window – a method that introduces different divisions of the raw recordings to generate additional data without interfering with the natural structure of the data, the size of the time frame is defined by the user.

- Savgol filter – a method that applies a Savitzky-Golay filter, to smoothen the IMU data.

- Add MLP output – a method which concatenates the MLP output into the pre-processed data, there is a possibility to create a dataset solely from the MLP output.

After using the chosen methods, the script inserts the data into a Data-loader object, for training and testing purposes. In addition, the scaling data, normalization scaler data, PCA and z score data are being saved for future use. Then, the data is loaded by a plotting script, which plot's the various features, both IMU and markers, for visual inspection. The next script in order is the model optimization script, it takes the data-loader files and create studies which try to optimize the model's test loss score. A study generates 1000 trials, each trial is a combination set of hyper-parameters to optimize. The script plots several parameters and scores of each model, train and test loss as a function of epoch, error heat-maps for specific recording matrix cell and specific time step, mean error as a function of time step, predicted wrist position vs. true wrist position 3D graph.

4.1.5 Live use script

This script connects to the Arduino and create a live input data stream. It uses the pre-processing methods for each data type (the data inserted into the MLP and the data inserted into the LSTM) it than loads both models, MLP and LSTM in evaluate mode, and passes the pre-processed data through them. It takes the resulting LSTM output prediction and broadcasts it to the robotic system, using sockets.

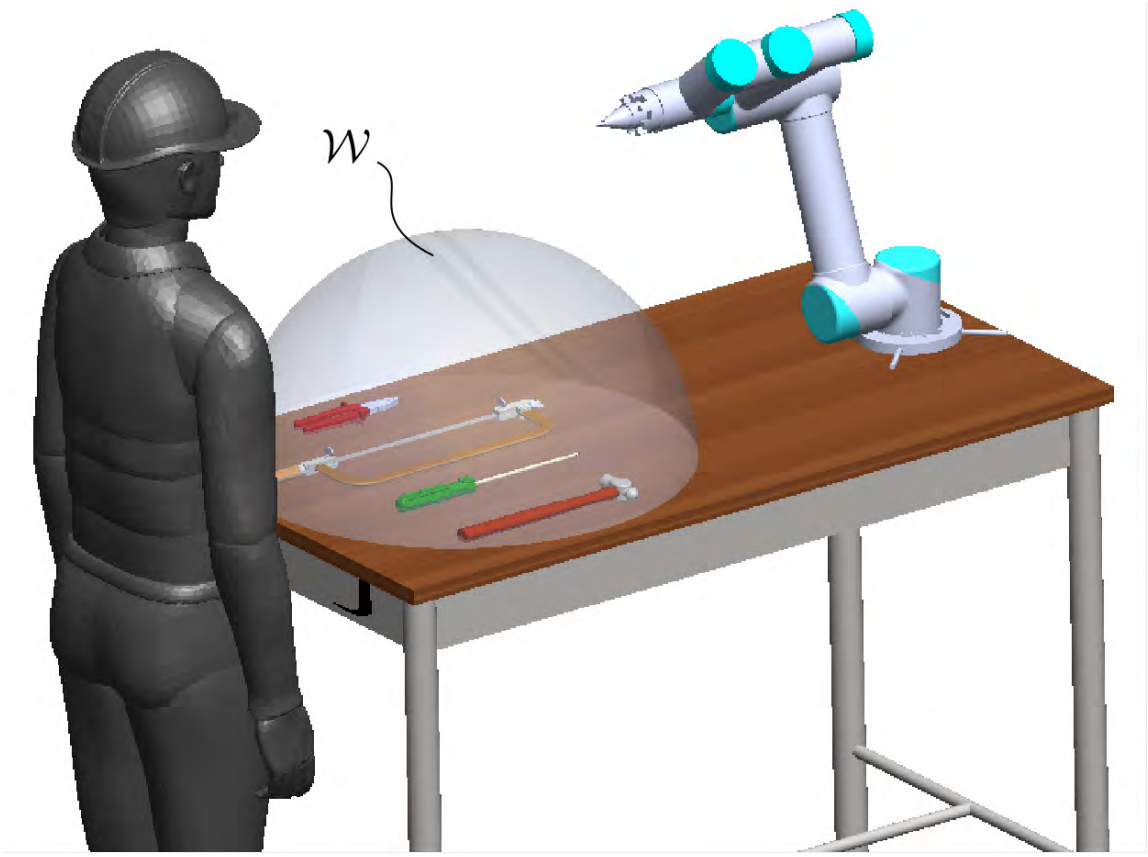

Figure 4.1: Human user and robot share space $\mathcal{W}$ in which they are to interact or collaborate in a shared task.

4.2 Learning human motion

4.2.1 Problem formulation

A human user stands in front of a designated interaction-space $\mathcal{W} \in \mathbb{R}^3$. Interaction-space $\mathcal{W}$ is shared with a robotic arm such that both user and robot can reach all points within it (Figure 4.1). Let the position of the human wrist at time t be $\mathbf{p}(t) \in \mathbb{R}^3$ relative to some coordinate frame on $\mathcal{W}$. At time $t = 0$, the wrist is at some arbitrary pose. The initial pose is randomly selected close to the body of the user (e.g., side of the hip or palm on the chest). The user then moves his arm to reach some point $\mathbf{p}(t_f) \in \mathcal{W}$ where $t_f < T$ for a pre-defined upper-bound T.

Let $\mathbf{x}(t) \in \mathbb{R}^n$ be the state vector of some measurable features on the human arm and denote $\tilde{\mathbf{p}} \in \mathbb{R}^3$ to be the estimated position of the wrist at time t. In addition, we denote $\tilde{\mathbf{p}}_f(t) \in \mathcal{W}$ to be the estimated destination of the wrist approximated at time $t < T$. We set two goals. First, we aim to localize the human wrist in real-time during motion. Hence,

we search for a map $\Gamma : \mathbb{R}^n \to \mathbb{R}^3$ that approximates the wrist position $\tilde{\mathbf{p}}(t) = \Gamma(\mathbf{x}(t))$ at time t. Second, we aim to learn a map $\Phi : \mathscr{C} \to \mathscr{W}$ that provides a prediction at time t of the expected wrist target at some time t_f. $\mathscr{C}$ is some unknown space based on measurable features in $\mathbf{x}(t)$ that can provide accurate prediction. In this work, we explore the formulation of $\mathscr{C}$ to acquire a sufficient representation of $\tilde{\mathbf{p}}_f(t) = \Phi(\mathbf{c}(t))$ for $\mathbf{c}(t) \in \mathscr{C}$.

4.2.2 System

We have designed and fabricated an experimental wearable device.

4.2.3 Electrical scheme and sensors

The system contains these electrical components:

1. Arduino uno controller, which is the used as a hub for all the sensors and a data acquisition device.

2. Multiplexer which switches between the IMUs.

3. Two Pmod-NAV IMUs which lay on the wrist and elbow, they measure the inertial forces exerted on the wrist and elbow, the angular velocity of the wrist and elbow and the change in the wrist and elbow in relation to the earth's megnetic field. All are measured in Cartesian 3D coordinate system. Hence, two IMU's provide a state vector $\mathbf{x}$ with maximum size of $n = 18$. We note that the measurements of a magnetometer are dependent on the relative orientation to earth's magnetic field. Hence, we assume that all recordings are taken such that the interaction-space is at the same orientation relative to the standing position of the human. Nevertheless, having another magnetometer on the human torso could enable relative measurements and additional interaction-spaces around the user.

4. USB type B cable, supplies power from the computer to the Arduino and transfers the raw data between the devices.

5. Personal computer, the data which is being acquired by the system is being fed in real time to a computer, which pre-process the data and feeds it into the model.

4.2.4 *Mechanical design*

The system is seen in Figure 4.2 and contains three main mechanical components:

1. Wrist 3D printed TPU band which lays on the wrist and a PLA cover which house the markers, the band contains an IMU.

2. Elbow 3D printed band which lays on the upper arm, contains a second IMU near the elbow and another marker configuration. In addition, contains the Arduino uno controller and the electrical components.

3. Shoulder 3D component which contains a marker configuration (for analysis and verification while not used in the modeling).

All of the printed components are designed to fit a large range of potential users, while attempting to maximize comfort, to avoid influence and change over the user's natural movement. In addition, the design is robust enough to withstand a large amount of repetitive use, due to the need of recording multiple time to create large data sets.

The acquisition configuration provides real-time data stream of all the given sensors in a frequency of 60 Hz. The reflective markers are tracked using a motion capture system and provide the positions of the shoulder, upper-arm and wrist bands. We note that the marker set is used only for data collection and validation while not required in the eventual system usage. Hence, the described system is composed of low-cost and light-weight (325 g) hardware which is appealing and suited for easy arm movements.

4.2.5 *Data collection and formation*

Data is collected over a set of K episodes. In each episode, the user starts from $\mathbf{p}(0)$ and moves the arm to $\mathbf{p}(t_f) \in \mathscr{W}$. During motion of episode j, states $\mathscr{X}_j = \{\mathbf{x}_j(0), \ldots, \mathbf{x}_j(t_f)\}$ and wrist positions $\mathscr{P}_j = \{\mathbf{p}_j(0), \ldots, \mathbf{p}_j(t_f)\}$ are recorded. For training model Γ, data from all episodes is pre-processed to have a labeled training dataset $\mathscr{M} = \{(\mathbf{x}_i, \mathbf{p}_i)\}_{i=1}^{N}$. The formation of the training data for target prediction will be discussed later.

4.2.6 *Wrist position model*

Let θ_{elv} and θ_{yaw} be elevation and yaw angles of the upper-arm, respectively. Similarly, let ϕ_{elv} and ϕ_{yaw} be the forearm elevation and yaw angles, respectively. The position of

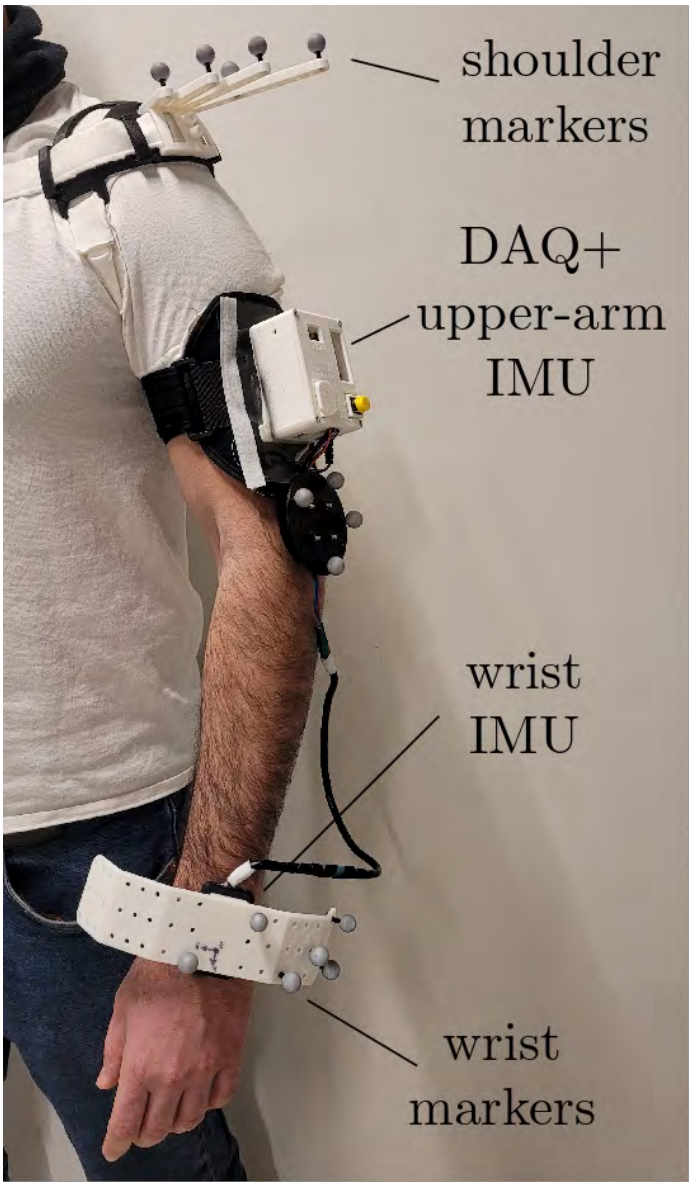

Figure 4.2: System of two IMU positioned on the wrist and upper-arm along with three sets of markers on the shoulder, upper-arm (not used) and wrist, observed by a motion capture system.

the wrist $\mathbf{p} = (p_x, p_y, p_z)^T$ is acquired by forward kinematics as given by Soechting and Flanders [55]:

$$p_x = l_u \sin\theta_{elv} \sin\theta_{yaw} + l_f \sin\phi_{elv} \sin\phi_{yaw} \tag{4.1}$$

$$p_y = l_u \sin\theta_{elv} \cos\theta_{yaw} + l_f \sin\phi_{elv} \cos\phi_{yaw} \tag{4.2}$$

$$p_z = -l_u \cos\theta_{elv} + l_f \cos\phi_{elv}. \tag{4.3}$$

where l_u and l_f are the lengths of the upper-arm and forearm, respectively. Assuming l_u and l_f are known, expressions (4.1)-(4.3) show that the position of the wrist can be acquired by measuring the orientations of the upper-arm and forearm. While these orientations can be measured by the accelerometers of the two IMU's when in a static pose, the two magnetometers can also do so during arm motion. Nevertheless, acceleration can add viable information for reaching $\mathscr{W}$.

Given training dataset $\mathscr{M}$, we train a feed-forward ANN to obtain map $\tilde{\mathbf{p}}_j = \Gamma_\theta(\mathbf{x}_j)$. Vector θ consists of the trained parameters of the model. In such case, user lengths l_u and l_f are embedded in $\mathscr{M}$ and explicit measurements are not required. Furthermore, the features to be included in state $\mathbf{x}_j$ and its size n such that Γ_θ achieves highest accuracy would be analyzed in the experimental section.

4.2.7 Long-Short Term Memory (LSTM)

LSTM is a class of Recurrent Neural-Networks (RNN) aimed to learn sequential data [68]. RNN utilizes previous outputs as inputs while including hidden states. For each time step t, the hidden state vector $\mathbf{h}(t)$ and the output $\mathbf{y}(t)$ are expressed as

$$\mathbf{h}(t) = g_1(W\mathbf{h}(t-1) + U\mathbf{x}(t) + \mathbf{b}_h) \tag{4.4}$$

and

$$\mathbf{y}(t) = g_2(V\mathbf{h}(t) + \mathbf{b}_y) \tag{4.5}$$

where W, U, V are weight matrices and $\mathbf{b}_h, \mathbf{b}_y$ are bias vectors. g_1 and g_2 are activation functions. The standard RNN is usually not capable in handling long intervals where back-propagating errors tend to vanish or explode [7]. LSTM, on the other hand, is capable of learning long-term dependencies by utilizing memory about previous inputs for an extended time duration [25]. Along with an hidden state vector, LSTM maintains a cell state vector $\mathbf{c}(t)$. At each time step, the process may choose to read from $\mathbf{c}(t)$, write to it or reset the cell using an explicit gating mechanism. Each cell unit has three gates

of the same shape. The input gate controls whether the memory cell is updated or, in other words, which information will be stored. An LSTM cell can be formulated with the following expressions:

$$i(t) = \sigma(W_i[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_i) \tag{4.6}$$

$$f(t) = \sigma(W_f[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_f) \tag{4.7}$$

$$o(t) = \sigma(W_o[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_o) \tag{4.8}$$

where W_i, W_f and W_o are weight matrices. $\mathbf{b}_i$, $\mathbf{b}_f$ and $\mathbf{b}_o$ are bias vectors. Forget gate $f(t)$ controls whether the memory cell is reset and removes irrelevant information from the cell state. Similarly, output gate $o(t)$ controls whether the information of the current cell state is made visible and adds useful information to the cell state. Both gates have a Sigmoid activation function σ. To modify the cell state, another vector $\tilde{\mathbf{c}}(t)$ is defined as

$$\tilde{\mathbf{c}}(t) = \tanh(W_c[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_c) \tag{4.9}$$

where W_c and $\mathbf{b}$ are weight matrix and bias vector, respectively. The hyperbolic activation function tanh distributes gradients and, therefore, prevents vanishing or exploding gradients, and allows a cell state information to flow longer. Vector $\tilde{\mathbf{c}}(t)$ is a new candidate that can be applied to the cell state in case the forget state chooses to reset. Hence, the new cell state $\mathbf{c}(t)$ is updated with

$$\mathbf{c}(t) = f(t)\mathbf{c}(t-1) + i(t)\tilde{\mathbf{c}}(t). \tag{4.10}$$

Furthermore, the hidden state is updated according to

$$\mathbf{h}(t) = \tanh(\mathbf{c}(t)) \cdot o(t). \tag{4.11}$$

The LSTM is trained using recorded data sequences with back-propagation.

4.2.8 Target prediction Model

Prediction based on acceleration data requires the integration of the signals along with position information. However, the motion is non-linear and an analytical representation is not available. Consequently and in addition to the sequential nature of the data, we require learning the motion pattern over some period. As noted above, LSTM has the ability to model sequences by selectively remembering certain patterns over some period and learn long-term dependencies. Therefore, we utilize LSTM to explore the various

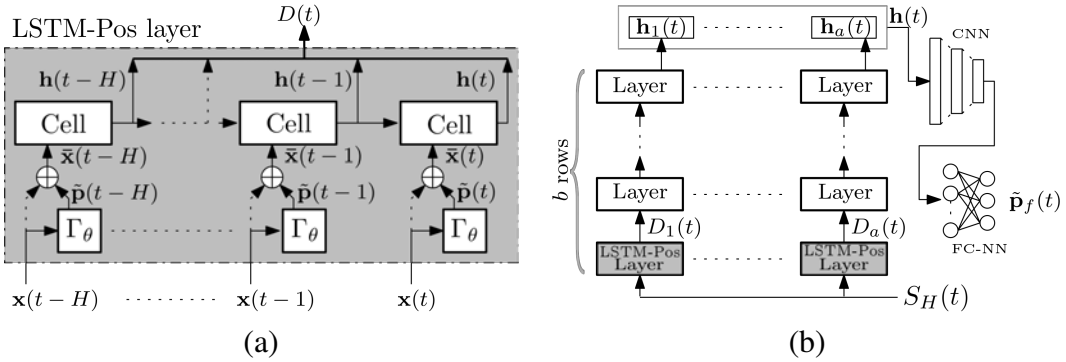

Figure 4.3: (a) A scheme of an LSTM-Pos layer receiving a state sequence $S_H(t) = \{\mathbf{x}(t-H), \ldots, \mathbf{x}(t)\}$ of length H with output vector $\mathbf{h}_j(t)$. The requirement for including the raw data with concatenation (denoted with symbol $\oplus$) would be further analysed. (b) The proposed LSTM-Pos network with stacked layers followed by a feed-forward output layer. The network outputs the predicted target of the wrist $\tilde{\mathbf{p}}_f(t)$ at time t.

feature and data size requirements in order to predict targets.

Let $S_H(t) \in \mathbb{R}^n \times \ldots \times \mathbb{R}^n$ be the sequence of H past states up to time t and is given by

$$S_H(t) = \{\mathbf{x}(t-H), \ldots, \mathbf{x}(t-1), \mathbf{x}(t)\}.$$

From each episode $\{\mathscr{X}_j, \mathscr{P}_j\}$, we extract sequences $S_H(H), S_H(H+1) \ldots S_H(t_f)$ and their respected target label $\mathbf{p}(t_f)$. Label $\mathbf{p}(t_f)$ is the last component of $\mathscr{P}_j$. Consequently, we acquire a training dataset from all episodes $\mathscr{L} = \{(S_{H,i}, \mathbf{p}_{f,i})\}_{i=1}^{M}$ where $S_{H,i}$ and $\mathbf{p}_{f,i} \in \mathscr{W}$ are the i^{th} sequence and target label, respectively, in the dataset.

Using dataset $\mathscr{L}$, we can directly train an LSTM model for Φ to predict wrist target. In such case, the input to the LSTM would be a sequence of states where each state has a maximum dimension of $n = 18$. In the experimental section, we will further investigate the importance of IMU features to the accuracy of the prediction. Nevertheless, we hypothesize that the corresponding estimation (using trained model Γ_θ) of wrist positions included in the state sequence would provide viable information for better accuracy. This is analogous to including initial conditions when integrating acceleration. Hence, we consider two alternatives for the state input $\bar{\mathbf{x}}(t-k)$ to the LSTM. In the first, we would feed only the approximated wrist positions to the LSTM

$$\bar{\mathbf{x}}(t-k) = \Gamma_\theta(\mathbf{x}(t-k))^T. \tag{4.12}$$

Alternatively, we observe the concatenation of the wrist position and the original raw data.

Hence, for each state in input sequence $S_H(t)$, we concatenate the approximated wrist position to generate a new state $\bar{\mathbf{x}}(t-j)$ given by

$$\bar{\mathbf{x}}(t-k) = \left(\mathbf{x}^T(t-k), \Gamma_\theta(\mathbf{x}(t-k))^T\right)^T. \tag{4.13}$$

Consequently, the input to the LSTM will now be the sequence along with the corresponding approximated wrist position. We would analyze these two alternatives in the experimental section.

We denote LSTM with wrist position information as *LSTM-Pos*. The architecture of an LSTM-Pos layer is seen in Figure 4.3a. The original state sequence $S_H(t)$ is the input to the layer followed by concatenation (4.13) using trained model Γ_θ. When considering state representation (4.12), the concatenation does not occur and approximated wrist positions are fed directly into the cells. A cell in the layer receives vector $\bar{\mathbf{x}}(t-k)$ and outputs hidden state vector $\mathbf{h}(t-k) \in \mathbb{R}^m$ where m is an hyper-parameter. Hidden state vectors are passed between the cells and additionally collected to hidden state sequence $D(t) = \{\mathbf{h}(t-H), \ldots, \mathbf{h}(t)\}$. Sequence $D(t)$ is the output of the LSTM-Pos layer.

Figure 4.3b illustrates the entire LSTM-Pos network. LSTM-Pos has b rows while each row has a layers. a and b are hyper-parameters of the network. Only the layers of the first row are LSTM-Pos layers where each outputs an hidden state sequence $D_j(t)$ for $j = 1, \ldots, a$. The other layers are regular LSTM layers that receive hidden state sequences and outputs updated ones. While all layers in a row have similar architecture, they include a random dropout such that $D_i(t) \neq D_j(t)$ for any $i, j \in \{1, \ldots, a\}$ and $i \neq j$. Furthermore, only hidden states at time t are outputted from the last row. Therefore, a hidden states $\mathbf{h}_1(t), \ldots, \mathbf{h}_a(t)$ are used and are the input to a standard Convolutional Neural-network (CNN) and further to a fully-connected NN (FC-NN). The FC-NN outputs the approximated target wrist position $\tilde{\mathbf{p}}_f(t)$ predicted at time t. In preliminary testings, training the model without a CNN failed to converge and is, therefore, essential. In the experimental section, we compare the LSTM-Pos to the standard LSTM which has the same architecture while not including wrist position information.

4.2.9 Curriculum learning

Preliminary testing has shown that training the models directly with all data may lead to predictions converging to the geometric center of the workspace. Hence, we use *Curriculum Learning* (CL) [61]. The common approach is to introduce the model with data in an organized order from easy samples to hard ones. Such approach commonly provides

better performance than random data shuffling.

We use preliminary insights on model performance to tackle the problem of convergence to a local minima. We propose to take a reverse CL approach and train with harder samples first. In preliminary studies, we have noticed that it is harder to predict the wrist pose and target at early stages of the reaching motion. This is due to low data variance when initiating motion. Hence, we train a model with N_{cl} batches along the motion starting from time $t = 0$. Once the loss value of the model successfully reaches below some user-defined value γ_{cl}, the next batch of further time frame is added to the training. When optimizing the hyper-parameters of some model, if value γ_{cl} is not reached for some batch, the model is disqualified.

Figure 4.4: Participant 1 collecting data of arm reaching in various torso orientations including facing the collection board, body perpendicular to the board and while sitting down.

4.3 Experiments

We use the experimental system described in Section 4.2.2 to collect data and analyze the proposed approach. Collection was performed by one human participant (participant 1) with the system on his left arm. Arm lengths can be seen in Table 4.3. To achieve uniform and guided data collection, a rectangular collection board was mounted in front of the participant as seen in Figure 4.4. The board was equally divided into 42 squares. The participant was asked to start from arbitrary arm location near the body and reach each of the squares several times while touching random locations within each square. To acquire a robust model, the participant also removed the system and re-positioned it on the arm several times during recordings.

A total of $K = 840$ reaching episodes were collected with 20 episodes for each square. Each episode was recorded in 60 Hz for $T = 2$ *sec* leading to 120 samples for an episode. Therefore, a set of $N = 100,800$ samples are available for training a wrist position model. The participant recorded reaching tasks with various torso orientations including facing the collection board, body perpendicular to the board and while sitting down. Hence, a large distribution of shoulder motions during arm reaching can be seen in Figure 4.5 showing high variance perturbations. A large testing dataset was also collected independent of the training set and includes 336 episodes when reaching eight episodes for each square in various torso poses. The test set also included removing and re-positioning the device between episodes in order to test robustness. We next evaluate the training wrist position and target prediction models with the data.

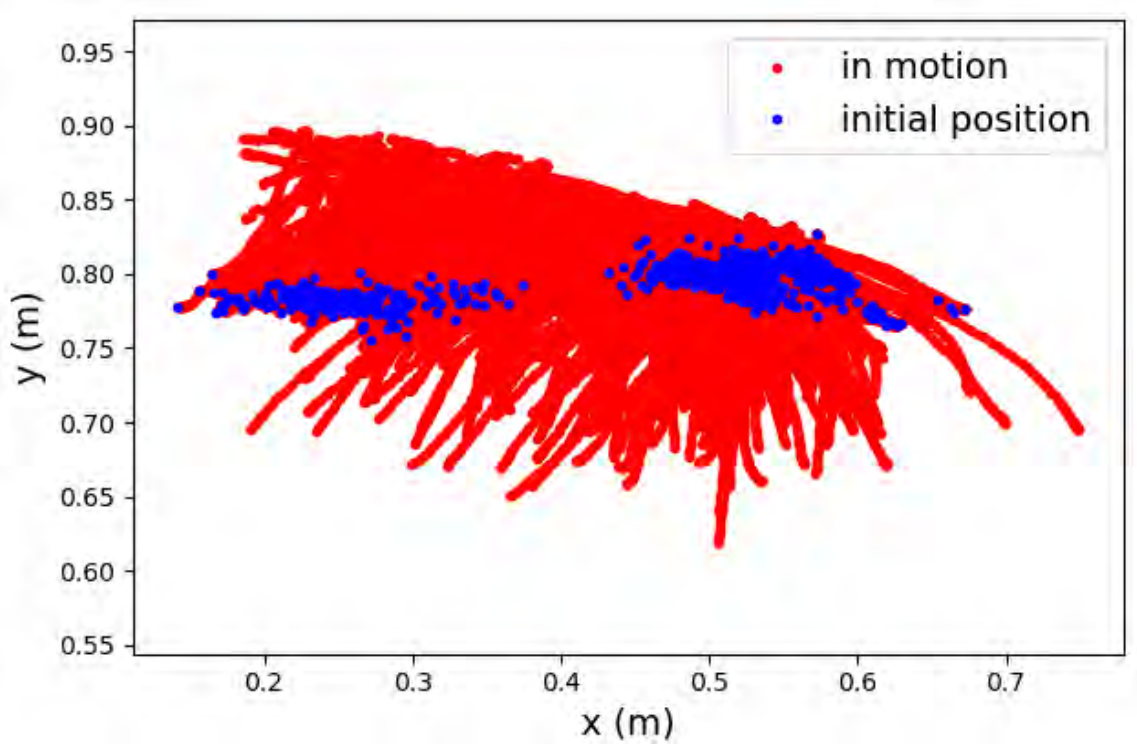

Figure 4.5: Shoulder positions relative to the board (at $y = 0$) before initiating motion (blue) and during motions (red).

Table 4.1: Approximation accuracy of wrist position during reaching motion

	w/o CL (mm)	w/ CL (mm)
All sensors	58.24 ± 4.79	48.62 ± 5.07
Accelerometers	99.71 ± 18.43	63.78 ± 14.73
Accel & Magn.	76.1 ± 5.63	58.81 ± 5.83
Gyroscopes	144.08 ± 17.24	124.79 ± 20.32
Magnetometers	93.79 ± 11.87	65.55 ± 8.43
Wrist only	64.51 ± 5.19	55.76 ± 5.86
Upper-arm only	149.1 ± 14.39	59.78 ± 10.8

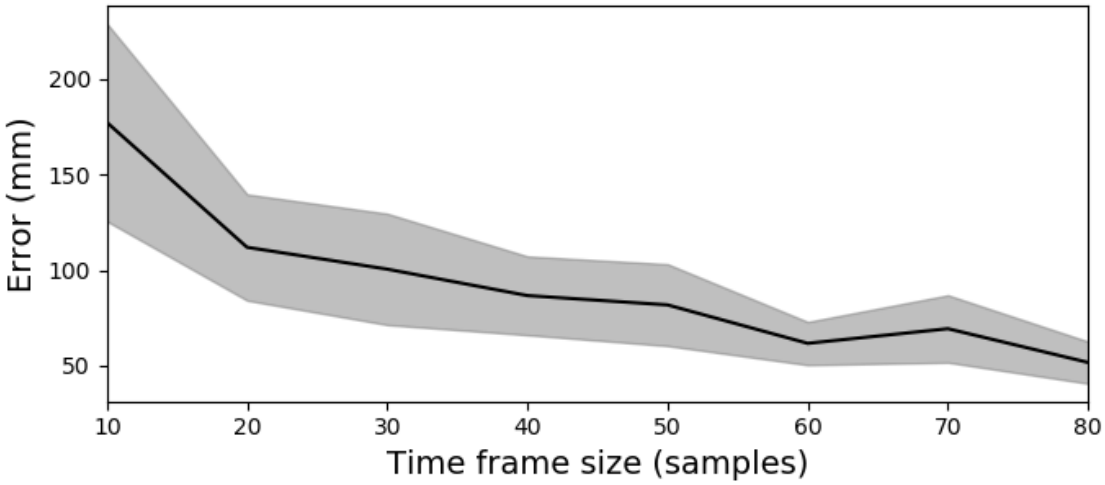

Figure 4.6: Target prediction error (mean and standard deviation) with respect to the number of the past states H.

4.3.1 Wrist position prediction evaluation

We now analyse the wrist position model Γ as discussed in Section 4.2.6. We analyze the accuracy of various ANN models trained with different features in the data. We test the ability of a model to predict the wrist position if only some of the features are available including: only accelerometers, only magnetometers, only gyroscopes, only accelerometers and magnetometers, only wrist IMU band, only upper-arm IMU band or all available sensors. For each variation, we optimized the hyper-parameters of the ANN to minimize the RMSE loss function along with the Adam optimizer. For example, the optimal architecture for the model with all sensing features included three hidden layers with 512 neurons each and a rectified linear unit (ReLU) activation function. We also compare between direct and CL training of the ANN. For the CL, we manually optimized the batch size and loss threshold yielding $N_{cl} = 10$ and $\gamma_{cl} = 58$ *mm*, respectively.

Table 4.1 summarizes the accuracy results for all variations. It is clear that CL provides accuracy improvement in all feature representations. Magnetometers by themselves supposedly should provide enough information to localize the arm. However and due to variations in torso orientations, only magnetometers provide moderate accuracy. Only using accelerometers reaches similar accuracy as they are able to encapsulate motion flow patterns but lack orientation information. Evidently, combining accelerometers and magnetometers provides accuracy improvement. Furthermore, using all available sensors of both IMU's (including the gyroscopes) provides the lowest error. The results also show

Table 4.2: Target prediction accuracy for LSTM and LSTM-Pos

Features	LSTM		LSTM-Pos	
	Mean (mm)	Std. (mm)	Mean (mm)	Std. (mm)
All sensors	194.63	73.07	61.46	11.26
Accelerometers	195.42	70.03	66.27	12.42
Accel. and Magn.	195.37	69.96	65.53	12.56
Gyroscopes	195.41	70.02	71.58	14.04
Magnetometers	195.43	69.99	72.71	19.12
Approx. wrist position	-	-	59.99	12.01
Wrist only	195.39	69.99	67.56	13.9
Upper-arm only	195.42	70.03	69.04	17.26

that having only one IMU either on the upper-arm or the wrist could provide fairly accurate wrist position. We note that these results are similar to results reported using the Unscented Kalman filter [4].

4.3.2 Target prediction evaluation

With the above NN models for approximating wrist position, we analyze the prediction of the target in reaching motions with the LSTM-Pos. We compare LSTM-Pos to an LSTM without including explicit approximation of wrist positions. Here also, we analyze the required features necessary to achieve accurate prediction. The hyper-parameters of models with different feature variations were optimized to yield the lowest RMSE loss value. For example, the optimal hyper-parameters for the model with all sensing features include LSTM with $a = 256$ and $b = 2$, $m = 64$, CNN with three convolutional layers and FC-NN with one hidden layer of 14 nodes, yielding 42,291 trainable parameters. In this part, all training is performed with CL since it provides better learning as demonstrated previously.

We initially observed the required number of past states H to acquire an accurate model. Figure 4.6 shows the mean prediction error over the test data when including all sensors and with regards to H. Including more information provides, as one would expect, better accuracy. Other feature variations exhibit similar behaviour. Having a large H means that the first prediction would arrive later in the reaching motion. Hence, we face a trade-off between earlier prediction and accuracy. In further analysis we choose to use $H = 60$ as it provides low error along with first prediction at half-way through-out the reaching motion. While not implemented in this work, these results show that one could train several models with ascending H. Hence, coarse accuracy would be provided at the

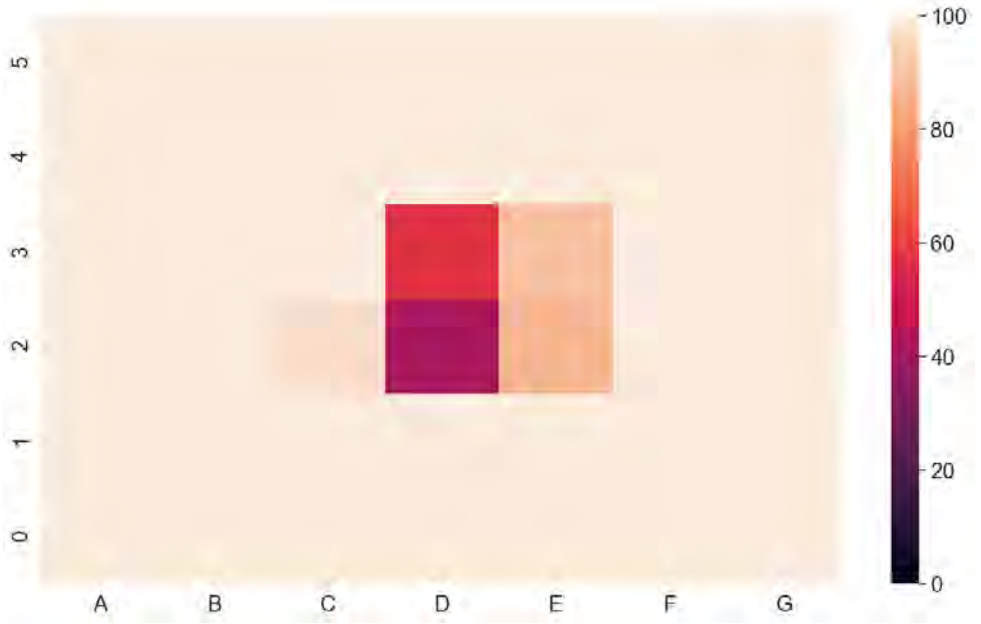

Figure 4.7: Typical Heat-map illustrating the mean error (in *mm*) across the collection board when not including wrist position approximation of LSTM-Pos. All models converged their prediction to the center of the board.

beginning of the episode (e.g., $H = 10$) for initial motion of the robot. Then, accuracy would be improved with better models as more information arrive.

Table 4.2 summarizes the target prediction errors (mean and standard deviation) along the reaching motion of all test data and for different feature variations. First, directly using the data to train an LSTM (without including wrist position approximation) fails to produce feasible predictions. All LSTM models converged to the mean of the target labels. Figure 4.7 shows an heatmap illustrating the error distribution across the collection board. Predictions for all test episodes output the center position of the collection board leading to low accuracy performance.

LSTM-Pos, on the other hand, provides much lower prediction errors for all feature variations. In particular, having only the sequence of approximated wrist positions from the NN fed into the LSTM is sufficient to acquire the best model. Including also the raw measurements from all sensors fairly provides the same accuracy. The results also indicate that having only one band can be sufficient for moderate accuracy. Figure 4.8 shows the mean error along the test episodes for LSTM-Pos with only approximated wrist positions and when including also raw sensor measurements (all sensors). Similarly, Figure 4.9 illustrates the mean errors across the collection board through the motion time. The above results achieve accuracy of approximately less than 70% of an adult human palm breadth

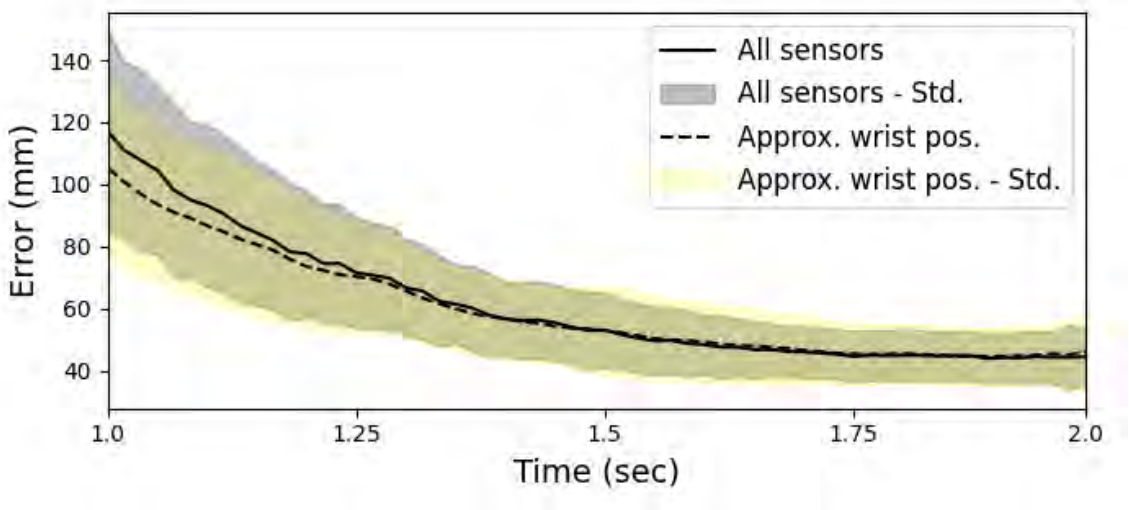

Figure 4.8: Target prediction error (mean and standard deviation) of LSTM-Pos with respect to reaching motion time evaluated on all test episodes.

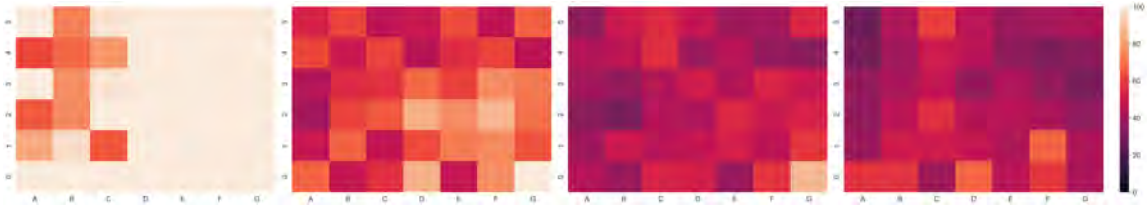

Figure 4.9: Heat-maps illustrating the mean error (in *mm*) across the collection board over time and when considering all sensors. Mean error from left to right: $Error(t = 1\ sec) = 116.98\ mm$**,** $Error(t = 1.33\ sec) = 62.41\ mm$**,** $Error(t = 1.65\ sec) = 46.86\ mm$ **and** $Error(t = 2\ sec) = 44.47\ mm$**.**

(87.5 mm) [6] and is sufficient for feasible HRC tasks as would be demonstrated in the next section.

4.3.3 HRC Demonstration

We have conducted an HRC demonstration where a participant reaches his arm towards a robotic arm. During the reaching motion, the robotic arm receives continuous stream of target predictions acquired from the trained model. When acquiring the first prediction, the robot plans motion for rendezvous and initiates motion. As a new prediction arrives, the motion plan of the robot is updated accordingly. Participant 1 along with two additional ones not included in training participated in the demonstration. The participants have different arm lengths as seen in Table 4.3. They were instructed to perform 15 reaching motions to arbitrary locations in front of them. They were also instructed to initiate motion

Table 4.3: Arm lengths of all participants

Participant	1	2	3
l_u (cm)	29	33.5	25
l_f (cm)	28.5	30.3	24.7

Table 4.4: Success rate for reaching experiment

Participant	Success rate
1	93.3%
2	80.0%
3	86.6%

from any desired arm pose. We define a successful trial as a one where the robot reached within the vicinity of the human hand with minimal distance of 60 mm.

Table 4.4 provides the rendezvous success rate for the three participants. The success for Participant 1 is the highest since the model is trained on data collected from him. Nevertheless, the two other participants achieved rather high success rates considering that they have different arm lengths and did not contribute data to model training. Hence, the model can be transferred to a new user with relatively good accuracy. Future work should consider calibrating the model to a new user with a small amount of collected data. Target wrist prediction was achieved in mid-motion and the robot initiated motion before the participant reached the end of the episode. For safety reasons, the velocity of the robot was maintained low. Hence, the participant was required to shortly wait for the arm. Nevertheless, having the robot move faster would enable faster HRC tasks.

Figures 4.10-4.12 show snapshots of some of the reaching trials for the three participants towards the robotic arm. Participant 3 in Figure 4.12 demonstrated the ability of the model to predict motion from various initial poses of the arm. Figure 4.13 shows a demonstration of handing over a bottle to the robot using the system. These results show that a relatively small dataset of IMU data can be used to train a sufficiently accurate model for feasible predictions of reaching targets in HRC scenarios. All experiments and demonstration videos can be seen in the supplementary video (Online Resource 1).

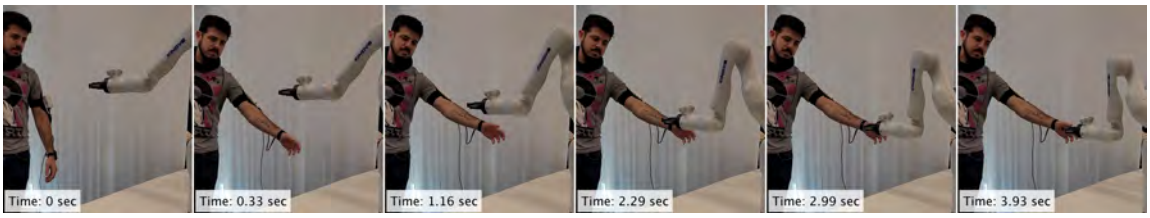

Figure 4.10: Reaching test of Participant 1.

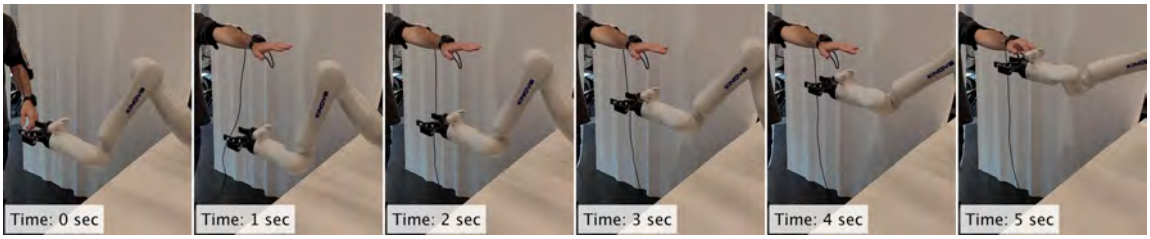

Figure 4.11: Reaching test of Participant 3.

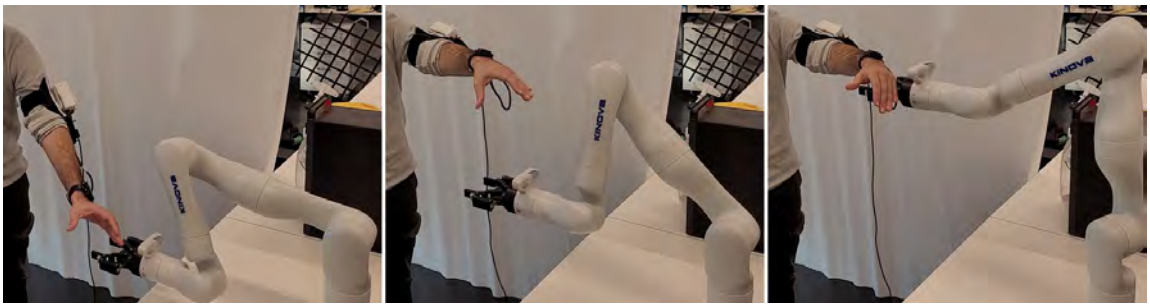

Figure 4.12: Motion of participant 2 between two arm poses within the interaction space while the robotic arm follows.

Figure 4.13: An HRC demonstration in which participant 1 hands-over a bottle to the robot.

4.4 Conclusions

We have presented the problem of learning human reaching motions and predicting future target locations. We investigated the use of a wearable device composed of two IMU senors located on the upper-arm and wrist. First, a simple ANN was trained to predict the current position of the wrist. Then, an LSTM-based architecture was proposed where approximation of the wrist position along some time frame was included. Using data collected from a single user, we have tested various features to be included in the model. Results show that having all features from the sensors along with curriculum learning achieve good accuracy for approximating current wrist position. With such model and without additional raw data, we reach sufficient accuracy for early prediction of the target wrist position. Furthermore, we have shown several reaching demonstrations where a robot planned and moved towards the human arm in real-time based solely on information from the wearable device. High success rates were demonstrated from the participant whom collected data and two other ones not included in the training. Hence, a relatively small amount of collected data from one user achieved sufficient accuracy for feasible predictions of reaching targets in HRC scenarios. Once trained, the model and system can be deployed in various spaces with no further effort.

Future work could focus on calibrating the existing model to a new user based on a limited amount of collected data. The system could also be fused with additional sensors. For instance, vision can be integrated in order to provide a complete solutions when a line-of-sight is not continuous. Similarly, one could integrate Force-Myography [27, 5] or EMG [8] to have additional information about the pose of the palm and fingers. An additional system on the other arm could enable the classification of tasks performed by the human towards assistance by a robot.

References

[1] Arash Ajoudani, Andrea Maria Zanchettin, Serena Ivaldi, Alin Albu-SchÃ€ffer, Kazuhiro Kosuge, and Oussama Khatib. Progress and prospects of the human-robot collaboration. *Autonomous Robots*, 10 2017. doi: 10.1007/s10514-017-9677-2.

[2] O. Amft, H. Junker, Paul Lukowicz, G. Troster, and C. Schuster. Sensing muscle activities with body-worn sensors. In *Int Work Wearable Implant Body Sens Networks*, pages 138–141, 05 2006. ISBN 0-7695-2547-4.

[3] Panagiotis Artemiadis. EMG-based robot control interfaces: Past, present and future. *Advances in Robotics and Automation*, 01, 2012. doi: 10.4172/2168-9695.1000e107.

[4] Arash Atrsaei, Hassan Salarieh, Aria Alasty, and Mohammad Abediny. Human arm motion tracking by inertial/magnetic sensors using unscented kalman filter and relative motion constraint. *Journal of Intelligent & Robotic Systems*, 90, 05 2018. doi: 10.1007/s10846-017-0645-z.

[5] Eran Bamani, Nadav D. Kahanowich, Inbar Ben-David, and Avishai Sintov. Robust multi-user in-hand object recognition in human-robot collaboration using a wearable force-myography device. *IEEE Robotics and Automation Letters*, 7(1):104–111, 2022. doi: 10.1109/LRA.2021.3118087.

[6] N. K. Bayraktar and E. Ozsahin. Anthropometric measurement of the hand. *Eastern Journal of Medicine*, 23(4):298–301, 2018.

[7] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[8] Luzheng Bi, Aberham Genetu Feleke, and Cuntai Guan. A review on EMG-based motor intention prediction of continuous human upper limb motion for human-robot collaboration. *Biomedical Signal Processing and Control*, 51:113 – 127, 2019. ISSN 1746-8094.

[9] Afonso Castro, Filipe Silva, and Vitor Santos. Trends of human-robot collaboration in industry contexts: Handover, learning, and metrics. *Sensors*, 21(12), 2021. ISSN 1424-8220. doi: 10.3390/s21124113. URL `https://www.mdpi.com/1424-8220/21/12/4113`.

[10] T. L. Chen, M. Ciocarlie, S. Cousins, P. M. Grice, K. Hawkins, K. Hsiao, C. C. Kemp, C. King, D. A. Lazewatsky, A. E. Leeper, H. Nguyen, A. Paepcke, C. Pantofaru, W. D. Smart, and L. Takayama. Robots for humanity: using assistive robotics to empower people with disabilities. *IEEE Rob. & Aut. Mag.*, 20(1):30–39, 2013. doi: 10.1109/MRA.2012.2229950.

[11] Erina Cho, Richard Chen, Lukas-Karim Merhi, Zhen Xiao, Brittany Pousett, and Carlo Menon. Force myography to control robotic upper extremity prostheses: A feasibility study. *Frontiers in Bioeng. and Biotech.*, 4, 2016.

[12] J. A. Corrales, F. A. Candelas, and F. Torres. Hybrid tracking of human operators using imu/uwb data fusion by a kalman filter. In *2008 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 193–200, 2008. doi: 10.1145/1349822.1349848.

[13] M. R. Cutkosky. On grasp choice, grasp models, and the design of hands for manufacturing tasks. *IEEE Transactions on Robotics and Automation*, 5(3):269–279, 1989. doi: 10.1109/70.34763.

[14] Cyberglove. Cyberglove. http://www.cyberglovesystems.com/, 2018.

[15] Hao Ding, Gunther Reibig, Kurniawan Wijaya, Dino Bortot, Klaus Bengler, and Olaf Stursberg. Human arm motion modeling and long-term prediction for safe and efficient human-robot-interaction. In *IEEE International Conference on Robotics and Automation*, pages 5875–5880, 2011. doi: 10.1109/ICRA.2011.5980248.

[16] T Flash and N Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of Neuroscience*, 5(7):1688–1703,

1985. ISSN 0270-6474. doi: 10.1523/JNEUROSCI.05-07-01688.1985. URL https://www.jneurosci.org/content/5/7/1688.

[17] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4346–4354, 2015.

[18] E. Fujiwara, Y. T. Wu, C. K. Suzuki, D. T. G. de Andrade, A. R. Neto, and E. Rohmer. Optical fiber force myography sensor for applications in prosthetic hand control. In *Proceedings of the IEEE International Workshop on Advanced Motion Control (AMC)*, pages 342–347, 2018.

[19] Nisal Menuka Gamage, Deepana Ishtaweera, Martin Weigel, and Anusha Withana. *So Predictable! Continuous 3D Hand Trajectory Prediction in Virtual Reality*, pages 332–343. Association for Computing Machinery, New York, NY, USA, 2021. ISBN 9781450386357.

[20] Brian Gleeson, Karon MacLean, Amir Haddadi, Elizabeth Croft, and Javier Alcazar. Gestures for industry intuitive human-robot communication from human observation. *sensors*, pages 349–356, 2013. doi: 10.1109/HRI.2013.6483609.

[21] Michael A Goodrich and Alan C Schultz. *Human-robot interaction: a survey*. Now Publishers Inc, 2008.

[22] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the International Conference on Machine Learning*, volume 70, pages 1321–1330, 2017.

[23] Christopher M. Harris and Daniel M. Wolpert. Signal-dependent noise determines motor planning. *Nature*, 394:780–784, 1998.

[24] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017.

[25] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[26] Xianta Jiang, Lukas-Karim Merhi, Zhen Gang Xiao, and Carlo Menon. Exploration of force myography and surface electromyography in hand gesture classification. *Medical Eng. & Physics*, 41:63 – 73, 2017. ISSN 1350-4533.

[27] Nadav Kahanowich and Avishai Sintov. Robust classification of grasped objects in intuitive human-robot collaboration using a wearable force-myography device. *IEEE Robotics and Automation Letters*, 6(2):1192–1199, 2021.

[28] Steven M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice-Hall, Inc., USA, 1993.

[29] Zeeshan Khokhar, Zhen Xiao, and Carlo Menon. Surface EMG pattern recognition for real-time control of a wrist exoskeleton. *Biomedical engineering online*, 9:41, Aug. 2010. doi: 10.1186/1475-925X-9-41.

[30] Markus Kohler. Using the kalman filter to track human interactive motion – modelling and initialization of the kalman filter for translational motion. Technical report, aaa, 1997.

[31] S. Kwon and J. Kim. Real-time upper limb motion estimation from surface electromyography and joint angular velocities using an artificial neural network for human-machine cooperation. *IEEE Transactions on Information Technology in Biomedicine*, 15(4):522–530, July 2011.

[32] Chiara Talignani Landi, Yujiao Cheng, Federica Ferraguti, Marcello BonfÃš, Cristian Secchi, and Masayoshi Tomizuka. Prediction of human arm target for robot reaching movements. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5950–5957, 2019. doi: 10.1109/IROS40897.2019.8968559.

[33] Przemyslaw A. Lasota, Terrence Song, and Julie A. Shah. *A Survey of Methods for Safe Human-Robot Interaction*. Foundations and Trends, 2017.

[34] Daniel Leal and Yimesker Yihun. Progress in human-robot collaboration for object handover. *Sensors*, pages C3–2–1–C3–2–6, 2019. doi: 10.1109/ISMCR47492.2019.8955665.

[35] Nan Li, Dapeng Yang, Li Jiang, Hong Liu, and Hegao Cai. Combined use of FSR sensor array and SVM classifier for finger motion recognition based on pressure distribution map. *Journal of Bionic Engineering*, 9(1):39–47, 2012.

[36] Ruixuan Liu and Changliu Liu. Human motion prediction using adaptable recurrent neural networks and inverse kinematics. *IEEE Control Systems Letters*, 5(5):1651–1656, 2021. doi: 10.1109/LCSYS.2020.3042609.

[37] Zitong Liu, Quan Liu, Wenjun Xu, Zhihao Liu, Zude Zhou, and Jie Chen. Deep learning-based human motion prediction considering context awareness for human-robot collaboration in manufacturing. *Procedia CIRP*, 83:272–278, 2019. ISSN 2212-8271. doi: https://doi.org/10.1016/j.procir.2019.04.080. URL https://www.sciencedirect.com/science/article/pii/S2212827119306948. 11th CIRP Conference on Industrial Product-Service Systems.

[38] Ruikun Luo, Rafi Hayne, and Dmitry Berenson. Unsupervised early prediction of human reaching for human-robot collaboration in shared workspaces. *Autonomous Robots*, 42:631–648, 2018.

[39] J. Martinez, M. J. Black, and J. Romero. On human motion prediction using recurrent neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4674–4683, 2017. doi: 10.1109/CVPR.2017.497. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.497.

[40] Toshiharu Mukai, Shinya Hirano, Morio Yoshida, Hiromichi Nakashima, Shijie Guo, and Yoshikazu Hayakawa. Tactile-based motion adjustment for the nursing-care assistant robot riba. *sensors*, pages 5435–5441, 2011. doi: 10.1109/ICRA.2011.5979559.

[41] Yasuhiko Nakanishi, Takufumi Yanagisawa, Duk Shin, Ryohei Fukuma, Chao Chen, Hiroyuki Kambara, Natsue Yoshimura, Masayuki Hirata, Toshiki Yoshimine, and Yasuharu Koike. Prediction of three-dimensional arm trajectories based on ecog signals recorded from human sensorimotor cortex. *PLOS ONE*, 8(8):1–9, 08 2013. doi: 10.1371/journal.pone.0072085. URL https://doi.org/10.1371/journal.pone.0072085.

[42] Heramb Nemlekar, Dharini Dutia, and Zhi Li. Object transfer point estimation for

fluent human-robot handovers. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2627–2633, 2019. doi: 10.1109/ICRA.2019.8794008.

[43] Joshua Newn, Benjamin Tag, Ronal Singh, Eduardo Velloso, and Frank Vetere. AI-mediated gaze-based intention recognition for smart eyewear: Opportunities and challenges. In *Proceedings of the ACM International Symposium on Wearable Computers*, pages 637–642, New York, NY, USA, 2019. ISBN 978-1-4503-6869-8.

[44] Luis Fernando Nicolas-Alonso and Jaime Gomez-Gil. Brain computer interfaces, a review. *sensors*, 12(2):1211–1279, 2012.

[45] Georg Ogris, Matthias Kreil, and Paul Lukowicz. Using FSR based muscule activity monitoring to recognize manipulative arm gestures. In *IEEE Int Symp Wearable Comput*, pages 45 – 48, 2007.

[46] Ozgur S. Oguz, Zhehua Zhou, and Dirk Wollherr. A hybrid framework for understanding and predicting human reaching motions. *Frontiers in Robotics and AI*, 5:27, 2018. URL `https://www.frontiersin.org/article/10.3389/frobt.2018.00027`.

[47] D. Paulius, Y. Huang, R. Milton, W. D. Buchanan, J. Sam, and Y. Sun. Functional object-oriented network for manipulation learning. In *IEEE/RSJ Int. Conf. on Intel. Robots and Sys.*, pages 2655–2662, 2016.

[48] Zhen Peng, Tim Genewein, and Daniel Braun. Assessing randomness and complexity in human motion trajectories through analysis of symbolic sequences. *Frontiers in Human Neuroscience*, 8:168, 2014. doi: 10.3389/fnhum.2014.00168. URL `https://www.frontiersin.org/article/10.3389/fnhum.2014.00168`.

[49] Claudia PÃ©rez-D'Arpino and Julie A. Shah. Fast target prediction of human reaching motion for cooperative human-robot manipulation tasks using time series classification. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6175–6182, 2015.

[50] Giacomo Rizzolatti, Leonardo Fogassi, and Vittorio Gallese. Neurophysiological mechanisms underlying the understanding and imitation of action. *Nature Reviews Neuroscience*, 2:661–670, 2001.

[51] Patrick Rosenberger, Akansel Cosgun, Rhys Newbury, Jun-Yung Kwan, Valerio Ortenzi, Peter I. Corke, and Manfred Grafinger. Object-independent human-to-robot handovers using real time robotic vision. *IEEE Robotics and Automation Letters*, 6: 17–23, 2021.

[52] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 683–700, Cham, 2020.

[53] P. B. Shull, S. Jiang, Y. Zhu, and X. Zhu. Hand gesture recognition and finger angle estimation via wrist-worn modified barometric pressure sensing. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 27(4):724–732, 2019.

[54] A. Singh, N. Thakur, and A. Sharma. A review of supervised machine learning algorithms. In *International Conference on Computing for Sustainable Global Development*, pages 1310–1315, 2016.

[55] J. F. Soechting and Martha Flanders. Errors in pointing are due to approximations in sensorimotor transformations. *Journal of neurophysiology*, 62 2:595–608, 1989.

[56] Yusuke Tamura, Masao Sugi, Jun Ota, and Tamio Arai. Prediction of target object based on human hand movement for handing-over between human and self-moving trays. In *ROMAN 2006 - The 15th IEEE International Symposium on Robot and Human Interactive Communication*, pages 189–194, 2006. doi: 10.1109/ROMAN.2006.314416.

[57] Mojgan Tavakolan, Zhen Xiao, and Carlo Menon. A preliminary investigation assessing the viability of classifying hand postures in seniors. *Biomedical engineering online*, 10:79, Sep 2011. doi: 10.1186/1475-925X-10-79.

[58] Yushuang Tian, Xiaoli Meng, Dapeng Tao, Dongquan Liu, and Chen Feng. Upper limb motion tracking with the integration of imu and kinect. *Neurocomputing*, 159:207–218, 2015. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2015.01.071. URL `https://www.sciencedirect.com/science/article/pii/S0925231215001472`.

[59] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[60] Yoji Uno, Mitsuo Kawato, and Ryoji Suzuki. Formation and control of optimal trajectory in human multijoint arm movement. *Biological Cybernetics*, 61:89–101, 1989.

[61] X. Wang, Y. Chen, and W. Zhu. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 1(01):1–1, 2021.

[62] Di Wu and Ling Shao. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 724–731, 2014. doi: 10.1109/CVPR.2014.98.

[63] Qingqiang Wu, Guanghua Xu, Min Li, Chen Longting, Xin Zhang, and Jun Xie. Human pose estimation method based on single depth image. *IET Computer Vision*, 12, 04 2018. doi: 10.1049/iet-cvi.2017.0536.

[64] Zhen Xiao and Carlo Menon. Towards the development of a wearable feedback system for monitoring the activities of the upper-extremities. *Journal of neuroengineering and rehabilitation*, 11:2, 01 2014. doi: 10.1186/1743-0003-11-2.

[65] J. Yang, K. Shen, C. Ong, and X. Li. Feature selection for mlp neural network: The use of random permutation of probabilistic outputs. *IEEE Trans. on Neural Net.*, 20 (12):1911–1922, 2009.

[66] Wei Yang, Chris Paxton, Arsalan Mousavian, Yu-Wei Chao, Maya Cakmak, and Dieter Fox. Reactive human-to-robot handovers of arbitrary objects. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3118–3124, 2021.

[67] H. K. Yap, A. Mao, J. C. H. Goh, and C. Yeow. Design of a wearable FMG sensing system for user intent detection during hand rehabilitation with a soft robotic glove. In *IEEE Int. Conf. on Biomedical Rob. and Biomech.*, pages 781–786, 2016.

[68] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Computation*, 31 (7):1235–1270, 07 2019.

[69] Xiaoping Yun and Eric R. Bachmann. Design, implementation, and experimental results of a quaternion-based kalman filter for human body motion tracking. *IEEE Transactions on Robotics*, 22(6):1216–1227, 2006. doi: 10.1109/TRO.2006.886270.

[70] Jianjing Zhang, Hongyi Liu, Qing Chang, Lihui Wang, and Robert X. Gao. Recurrent neural network for motion trajectory prediction in human-robot collaborative assembly. *CIRP Annals*, 69(1):9–12, 2020. ISSN 0007-8506. doi: https://doi.org/10.1016/j.cirp.2020.04.077. URL https://www.sciencedirect.com/science/article/pii/S0007850620300998.