Proportional and Simultaneous Real-Time Control of the Full Human Hand From High-Density Electromyography

Raul C. Sîmpetru, *Graduate Student Member, IEEE*, Michael März, and Alessandro Del Vecchio, *Member, IEEE*

***Abstract*— Surface electromyography (sEMG) is a non-invasive technique that measures the electrical activity generated by the muscles using sensors placed on the skin. It has been widely used in the field of prosthetics and other assistive systems because of the physiological connection between muscle electrical activity and movement dynamics. However, most existing sEMG-based decoding algorithms show a limited number of detectable degrees of freedom that can be proportionally and simultaneously controlled in real-time, which limits the use of EMG in a wide range of applications, including prosthetics and other consumer-level applications (e.g., human/machine interfacing). In this work, we propose a new deep learning method that can decode and map the electrophysiological activity of the forearm muscles into proportional and simultaneous control of > 20 degrees of freedom of the human hand with real-time resolution and with latency within the neuromuscular delays (< 50 ms). We recorded the kinematics of the human hand during grasping, pinching, individual digit movements and three gestures at slow (0.5 Hz) and fast (0.75 Hz) movement speeds in healthy participants. We demonstrate that our neural network can predict the kinematics of the hand in real-time at a constant 32 predictions per second. To achieve this, we employed transfer learning and created a prediction smoothing algorithm for the output of the neural network that reconstructed the full geometry of the hand in three-dimensional Cartesian space in real-time. Our results demonstrate that high-density EMG signals from the forearm muscles contain almost all the information that is needed to predict the kinematics of the human hand. The proposed method has the capability of predicting the full kinematics of the human hand with real-time resolution with immediate translational impact in subjects with motor impairments.**

***Index Terms*— EMG, real-time systems, kinematics, deep learning, transfer learning.**

I. Introduction

REAL-TIME, proportional, and detailed control of the human hand using the electromyographic (EMG) signal is a difficult challenge. Although the EMG signals represent the current state of the art in controlling prosthetics devices [1], [2], the number of degrees of freedom that can be decoded simultaneously and with real-time resolution in healthy humans is small (≤ 6), with poorer results in humans with amputations [3], [4], [5], [6], [7], [8] or spinal cord injury [9].

In recent years, incredible hardware advances have been made in hand prostheses [1], [10], [11], [12], [13], [14], [15], [16], [17], [18]. Despite being able to mechanically replicate the movements of human hands, the current state of movement intent detection systems hinders the full potential of these devices due to limitations in software and hardware that are needed to control such hands in a natural and intuitive way [1], [2], [18], [19]. This discrepancy is one of the main reason why hand prostheses are not yet widely accepted by the users [2], [19], [20], [21].

At the physiological level, the central nervous system generates a control signal which is sent to the muscles through the final pathway of movement: the spinal α-motor neurons [22]. The surface EMG [23], [24], [25] allows non-invasive access to a surrogate signal that is correlated with the sequences of motor neurons firings depicted by the recording electrodes as ensembles of muscle fiber action potentials [26], [27]. There has been significant research exploring the use of sEMG to augment the degrees of freedom of prostheses. Although there is a lot of work in decoding algorithms for myocontrol, only a small number of algorithms are able to make real-time predictions [3], [4], [5], [6], [7], [8], [28], which are crucial for real life applications. While sEMG signals can be highly informative, they can also be quite variable due to a variety of factors, including the specific muscles being activated, the intensity of the muscle contractions, neuromuscular fatigue, and the distinct recruitment patterns of motor units encod-

Manuscript received 12 January 2023; revised 26 May 2023 and 8 July 2023; accepted 10 July 2023. Date of publication 13 July 2023; date of current version 4 August 2023. This work was supported by the German Ministry for Education and Research (BMBF) through the Project MYOREHAB under Grant 01DN23002. *(Corresponding author: Alessandro Del Vecchio.)*

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by University of Erlangen-Nuremberg under Application No. 21-150_3-B, and performed in line with the Declaration of Helsinki.

The authors are with the Neuromuscular Physiology and Neural Interfacing (N-squared) Laboratory, Department Artificial Intelligence in Biomedical Engineering (AIBE), Friedrich-Alexander-Universität Erlangen-Nürnberg, 91052 Erlangen, Germany (e-mail: alessandro.del.vecchio@fau.de).

This article has supplementary downloadable material available at https://doi.org/10.1109/TNSRE.2023.3295060, provided by the authors.

Digital Object Identifier 10.1109/TNSRE.2023.3295060

ing specific hand tasks [23], [24], [25]. To overcome this variability all the current algorithms try to extract features that are mostly undisturbed by the mentioned factors, which mainly consist of low-pass filtering the EMG signals (< 20 Hz) and/or extracting features from a relatively low number of EMG signals. These factors can cause a significant loss of neural information that is needed to proportionally and simultaneously control a high number of degrees of freedom from extrinsic muscle activity. This results in at most 6 degrees of freedom that the users are capable of controlling [3], [4], [5], [6], [7], [8], [28] reliably. Most often this control takes the form of a cursor in 2D [6], [7], [8], [28] or as only one movement (grasping) in 3D [5].

Here, we demonstrate with real-time simultaneous and proportional control, that this problem is not caused by intrinsic problems associated with the EMG (i.e., that there is insufficient information contained in the signal itself) but rather by inadequate processing algorithms that are used to process the signal which is in a relatively large frequency range (20-500 Hz). In our prior conference study we investigated the offline movement decoding solutions with simulated real-time applications [29], and produced a neural network that was capable of accurate prediction on unseen data [29], [30]. That neural network is now adapted for real-time usage in this work (Fig. 1 shows an overview of this study) and is now able to predict the entire hand (21 joints) in Cartesian space at a constant 32 predictions per second. Our study demonstrates the ability to continuously decode and execute 12 hand and wrist movements, including individual finger flexion and extension, fist opening and closing, two-finger and three-finger pinching, wrist abduction and adduction, and maintaining three different hand postures. These movements were successfully decoded for 10 healthy participants at both fast (0.75 Hz) and slow (0.5 Hz) speeds, solely by analyzing the activity of the extrinsic muscles in the forearm.

This work may lay the foundation for how neural interfaces should be used to robustly and reliably decode human motion intentions for future human/machine interface applications. Moreover, our neural network also shows the capability to be widely used in neuroscience and physiology, since it is possible to understand the latent components learned by the AI and thus learn the contribution of different muscle compartments to specific hand postures in a direct way.

II. Methods

A. Participant Characteristics

We collected data from 10 subjects, 4 females and 6 males (Fig. 2(a)). Two subjects (3 and 4) were left-handed. The subjects in this study are young, healthy adults with an age of 25.10 ± 4.23 years and a weight of 74.10 ± 10.20 kg.

Two separate experiments were conducted in two distinct sessions, spaced several months apart. The first experiment involved nine participants (labeled subjects 1 through 9), comprising four females and five males. The objective of this experiment, referred to as the "movement-following" experiment, aimed to assess the feasibility of utilizing offline-trained neural networks for real-time kinematics prediction with minimal retraining required on different days.

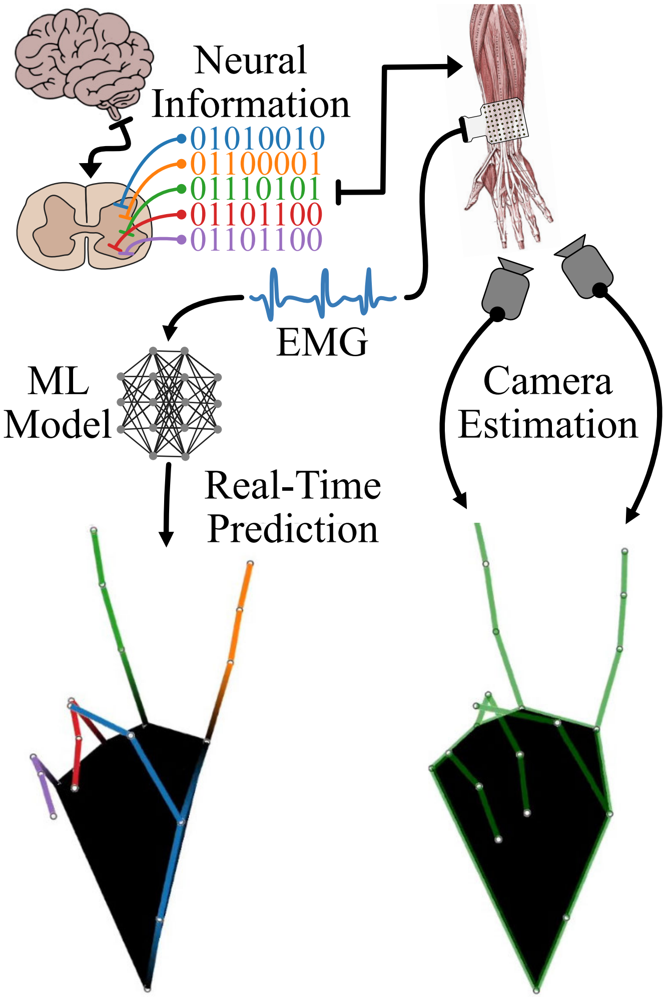


Fig. 1. In this study, we recorded sEMG data from ten subjects while they performed various hand movements. The subjects were shown videos of pre-recorded movements to help them maintain a consistent movement frequency. The recorded kinematics and sEMG were then used to train a deep learning model. In a separate session, the trained model was tested in real-time. We utilized transfer learning to adapt the model to the subject's new electrode grid placement, which can vary from recording to recording. This process took approximately 15 min. Real-time feedback was provided to the subjects through a virtual 3D hand rendering. The two hands displayed can be seen in Video 1.

The second experiment involved three subjects (subjects 1, 3, and 10), including one female and two males. Termed the "target reaching" experiment, the primary objective was to evaluate if the neural network would be an effective user-in-the-loop system. These participants were instructed to reach as many targets as possible within a 10 min timeframe while the time to reach was recorded.

All experiments were reviewed and approved by the ethics committee of the University of Erlangen–Nuremberg (application no. 21-150_3-B) for compliance with the Declaration of Helsinki, and the subjects signed an informed consent form.

B. Data Acquisition

The training dataset for our model was created by recording the sEMG signals from the muscles of the dominant hand of the participants (Fig. 2(a)) during movements performed synchronously with existing kinematics. Our first goal was

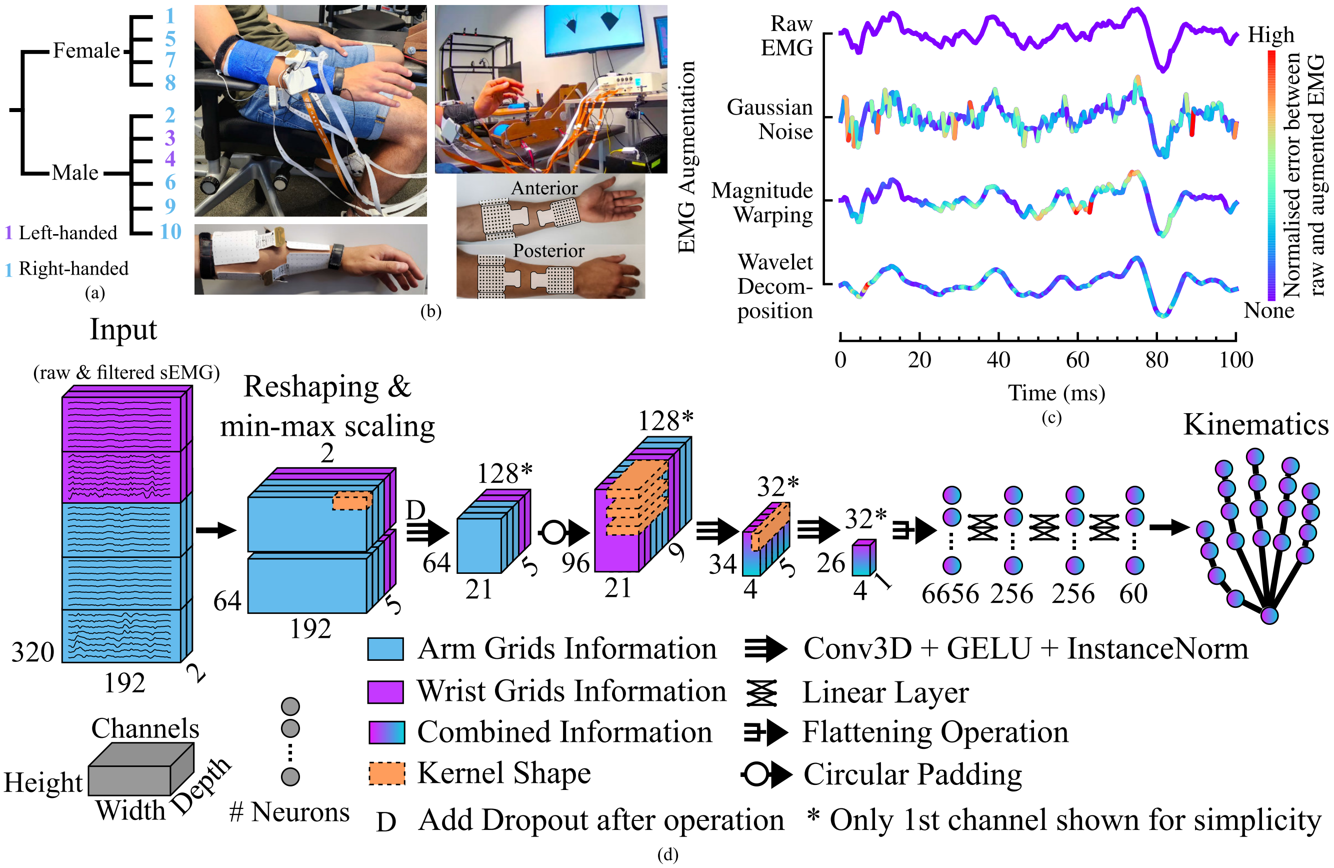


Fig. 2. (a) Dendrogram showing the specifics of each subject. We recruited a total of 10 subjects, 4 females and 6 males. Two of the males were left-handed. (b) Experimental setup and grid placement. Two 8 × 8 electrode grids are placed over the wrist and three 8 × 8s under the elbow, with a total of 320 electrodes. (c) Comparison between the raw EMG signal and three different augmentation methods [31], coloring the altered signals based on the normalized absolute difference from the raw signal. In short, *Gaussian Noise* [31, p. 3] modifies the signal to have a signal-to-noise ratio of 5, *Magnitude Warping* [31, p. 3] accounts for the shift in the electrode grids, and *Wavelet Decomposition* [31, p. 4] facilitates model generalization by reconstructing the original signal with noise. (d) Schematic overview of our adapted model from Sîmpetru et al. [30] for real-time inference. Each input grid displays 8 random electrode signals from that particular grid.

to extract natural three-dimensional hand joint positions in reference to the wrist in order to train the deep learning model (Fig. 1). Because we did not want to constrain the model to a user-specific hand skeleton or to a hard-coded skeleton, we recorded the ground-truth data from a single human individual that performed the full hand tasks (see below). All of the individuals were then asked during the experiments to mimic the movements of the 3D hand (Fig. 1, Video 1). For left-handed participants we mirrored the video horizontally. Left-handedness did not impact any other part of the experiments. We obtained the kinematics of subject 1 by having them perform the following movements 3 times each in our kinematics recording setup:

- flexion of each digit
- resting
- pinching between index and thumb (2-finger pinch)
- pinching between index, middle finger, and thumb (3-finger pinch)
- adduction and abduction of the wrist
- fist closing and opening
- pointing
- peace sign
- rock and roll sign

We recorded three hand gestures (pointing, peace sign, and rock sign) as well as dynamic movements in order to investigate whether the model would have an easier time decoding the gestures or the dynamic movements and if there would be any discrepancies for the model to switching from individual finger actions to gestures. The hand gestures have been chosen because they are easily recognizable and therefore performable by almost everyone without the need of pretraining. The acquisition system (detailed explanation in Cakici et al. [32]) used 4 cameras distributed around a modular frame. The cameras recorded the movement of the hand simultaneously from different angles. The resulting videos were processed by the markerless kinematics software DeepLabCut [33] and aligned in 3D space with Anipose [34] to obtain reliable 3D kinematics data (see Video 1 and Fig. 1-3).

Based on subject 1's kinematics, we created videos that assisted and guided the participants in maintaining a steady frequency while they performed the movements. These kinematics were also used as the ground truth because this allowed us to skip the acquisition of kinematics for all healthy subjects and can also allow subjects who cannot move their hand due to amputations or spinal cord injury to participate in the study in the future. This is an important prerogative of the study, as we did not want to use a virtual hand as an output signal but a physiological signal that also contains the enslaving features of a real human hand from a healthy individual.

The sEMG was acquired from 5 electrode grids (8 rows x 8 columns, 10 mm interelectrode distance; OT Bioelettronica, Turin, Italy). Fig. 2(b) shows the positioning of the electrodes. Three high-density EMG grids were placed around the thickest part of the forearm and two around the wrist, proximal to the ulnar head. Before placing the electrode grids, the skin was shaved and cleaned with an alcoholic solution. The placement of the electrode grids is designed to cover both the distal and proximal muscles of the forearm, thereby facilitating the acquisition of data from the majority of extrinsic muscles of the hand. However, it is important to acknowledge that the intricate overlap of multiple muscle groups in the forearm presented a considerable challenge in accurately pinpointing and specifically targeting individual muscles. Consequently, the primary emphasis of this study revolved around predicting kinematics without the reliance on recording specific muscles, resulting in a decreased requirement for extensive anatomical expertise. This, in turn, has the potential to broaden the audience that can effectively utilize our system.

The monopolar sEMG signals were recorded using a multichannel amplifier (EMG-Quattrocento, A/D converted to 16 bits; OT Bioelettronica, Turin, Italy), amplified (×150) and band-pass filtered (0.7–500 Hz) at source. The signals were sampled at 2048 Hz and captured during the display of the motion videos using a custom script written in Python. The EMG recorder was programmed to produce sEMG signals in 32 non-overlapping windows every second. When recording at a sampling rate of 2048 Hz, this setting resulted in 32 windows, each containing 64 samples of the raw monopolar EMG signal. At the same time, we also calculated the time stamps to which each segment corresponds in video frames. When the recording was finished, we used these intervals to store the synchronized kinematics with the sEMG signals.

For the movement-following experiment, each subject had to participate on 2 different days (except subject 1 who had one extra day for kinematics acquisition). On the first day, we recorded the EMG while the subjects follow the kinematics videos that guides the subjects to perform all the planned movement trajectories. The experimental setup can be seen in Fig. 2(b). We used 30 s per movement, resulting in a total of 12 min of data for training a subject specific model from scratch. On the second day (intervals between the days varied with a mean of a week between the two laboratory visits) the subjects were recorded again for 10 s per movement. This gave us 4 min of data in order to apply transfer learning (see below) to account for the signal variability due to the repositioning of the electrodes and day-to-day variability in the EMG signal. After the subject-specific model trains for about 8 min, we gave the subjects 15 min to familiarize themselves with the interface (lower part of Fig. 1, detailed explanation in section II-G). At the end of the experiment, we asked the subjects to try to mimic the movements of a guiding hand as closely as possible for 15 s per movement. We record the positions of both their virtual hand and the guiding hand for later analysis (see Video 1).

The subsequent target reaching experiment was conducted several months after the initial experiment and could be completed within a single day. Each movement during the experiment was recorded for a duration of 20 s, resulting in a total of 8 min of training data for constructing subject-specific models from scratch. The training process, utilizing 4 NVIDIA RTX 3080 GPUs (unavailable during the initial experiment), took approximately 30 min.

During the training phase, we introduced the subjects to the new interface (Fig. 7(a)), which featured a virtual hand and colored dots. The colored dots corresponded to the fingers' colors and represented the desired 3D positions that each finger should move to in order to reach the required target positions. A target was deemed successfully reached only if the user maintained an average distance of less than 20 mm between the target dots and their corresponding fingers for a duration of 2 s. In the event that this criterion was met or not met within a 10 s interval, a new target was selected, ensuring that consecutive targets were never identical.

Using this interface, we instructed the subjects to reach as many targets as possible within the given 10 min timeframe. To provide further guidance, short descriptions of the targets were displayed above the virtual hand, such as "Rock" for the rock sign (Fig. 7(a)). Additionally, the text changed color from red to green, serving as feedback to indicate whether the hand position was within the required 20 mm threshold.

C. Preprocessing

The sEMG signals are given in non-overlapping windows of 64 samples (31.25 ms) at a rate of 32 Hz. We found out that this did not provide enough temporal resolution for our system to work effectively, so we implemented a queue of length 3. This allowed us to wait for three sEMG segments to be provided, and then use these to create a longer 93.75 ms segment. We moved the queue one 31.25 ms segment at a time, which dropped the oldest chunk (the first in the queue) and added the newest (the third in the queue) at the end of it. This means that our system will have a warm-up time of 93.75 ms before being able to provide useful predictions. After this warm-up time, our system can display the three-dimensional hand predictions in 38.7 ms (31.25 waiting for the next EMG segment and 7.45 for the prediction and correction, Fig. 8). Therefore, the 38.7 ms latency represents the time for the system to make inferences about movement. This is a very small delay in comparison to other studies and most importantly it is in the range of the physiological electromechanical delays during hand digit movements, which range from 40 to 250 ms (see results on the neuromechanical delay in Vecchio et al. [35]).

We copied each longer segment and low-pass filtered it forwards and backwards with a 4th order digital Butterworth filter below 20 Hz. The filtered version was then appended to the raw (digitally unfiltered) segment in the depth dimension, resulting in an sEMG tensor of the shape *depth (raw or filtered)* × *number of electrodes* × *time in samples*. In previous offline experiments, we have shown that the optimal performance for a deep learning architecture, similar to the one employed in this study, was achieved by simultaneously inputting both the raw monopolar EMG signals and a low-pass filtered version (20 Hz) of those signal [29], [30].

D. EMG Augmentation

Recording each movement for 30 s was necessary in order to complete the data collection in a reasonable time frame. This however does not result in a lot of data for the deep learning model, therefore we developed an EMG augmentation pipeline with three different augmentation methods that can be applied to each EMG segment (Fig. 2(c), augmentations taken from Tsinganos et al. [31]). In the first method, called *Gaussian Noise* [31, p. 3], Gaussian noise is added to each EMG channel such that the signal-to-noise ratio between the unaugmented and augmented EMG segment of 5 is achieved. To account for possible displacement of the electrode grids on the skin, we apply *Magnitude Warping* [31, p. 3]. This is achieved by multiplying a curve sampled from normal distributions to the EMG channels. Finally, we apply *Wavelet Decomposition* [31, p. 4] to facilitate model generalization by reconstructing a distorted but similar EMG signal. To distort the signal we multiply a constant to the detail coefficients. Using the inverse wavelet decomposition we can extract the distorted signal. Taken together, these methods provide a 4× increase in data and allow us to prepare the model for problems that occur in real life.

E. Model

This work is a real-time adaptation of our previous works [29], [30], both based on the same basic model, which we updated to the latest available machine learning frameworks for Python (PyTorch [36] 1.12.0+cu116 and PyTorch-Lightning [37] 1.7.1). For this reason, we will explain our adjustments in detail, but keep the general description of the architecture short, as a detailed explanation can be found in [30]. A graphical overview of the architecture of the model is given in Fig. 2(d). The general architecture presented in this paper is that of a convolutional neural network [38, ch. 9] followed by a multilayer perceptron [38, ch. 6], albeit heavily optimized for real-time processing of EMG signals.

The inputs to the model are three dimensional EMG tensors. The width contains the time samples, the height contains all 320 electrodes (5 × 64), and the depth is used to store the raw and filtered information. The model reshapes the tensors by splitting the second dimension (total number of electrodes) into two dimensions: one for the number of grids we have and one for the number of electrodes per grid. The reshaped tensor then has the abstract shape *depth (raw or filtered)* × *number of grids* × *number of electrodes per grid* × *time in samples* or in actual numbers *2* × *5* × *64* × *192*. Using the reshaped tensor we applied the grid-wise normalization (this is different from the normalization in Sîmpetru et al. [30])

$$\tilde{x} = \frac{x - \text{mean}(x)}{\text{std}(x)} \tag{1}$$

where x is the original grid sEMG signal, mean() gives the mean, std() gives the standard deviation, and $\tilde{x}$ is the normalized sEMG signal.

The other adaption we undertook was to use *Instance Normalization* [39] instead of *Batch Normalization* [40]. The sEMG signal exhibits significant stochasticity and is susceptible to data drift resulting from factors such as grid positioning. Consequently, the statistical information (mean and standard deviation) acquired from batches are not indicative of the sEMG recording in real-time scenarios, leading to a decrease in the performance of the neural network. As normalization is still necessary for faster convergence we used *Instance Normalization* as this method computes the mean and standard deviation required for each input in the batch individually and efficiently. It is important to mention that we always compute normalization during testing, without using any statistics that were computed during training. This helped in making our system more robust and ensured that it could not have relayed on the training distribution.

We changed the activation function to GELU [41] as it is faster then the learnable activation functions we have used in our prior works [29], [30]. To further improve the inference time we have also reduced the number of channels per layer.

Since the placement of the electrode grids differed slightly between the original recording and the real-time tests of the movement-following experiment, we first trained subject-specific models from scratch and used transfer learning so that the networks would learn the new grid positions and, if necessary, learn which channels were faulty.

We optimized our models with the mean absolute error as the loss function using the AdamW [42] optimizer with the AMSGrad [43] correction. The weight decay was 0.01. For the initial training from scratch, we train for 50 epochs. However, for transfer learning, we only use 12 epochs. For training, we used the one-cycle approach described in Smith and Topin [44] as a learning rate scheduler with an upper bound of $10^{-2.5}$, lower bound 10^{-7} and initial learning rate 10^{-4} up to half of the epochs, after which we exchanged it for the stochastic weight averaging approach of Izmailov et al. [45] using cosine annealing for 5 epochs and a learning rate of 10^{-3}. We also use the same parameters for the one-cycle scheduler during transfer learning, but switch it after only 4 epochs for stochastic weight averaging and use only 3 instead of 5 epochs for cosine annealing.

The hyperparameter optimization was significantly influenced by our previous work where our model was first described [29]. In order to arrive at a reduced number of layers per channel and still maintain a reasonable loss, we used 10% of the data from subject 1 for the hyperparameter optimization. Since all tests are performed on a new day with new unseen data, there is no possibility of overfitting to the test data, however, due to stochasticity and the technical difficulty of performing large hyperparameter searches, the architecture could still be improved. The high variability of the EMG signal further adds to the difficulty of the optimization, however we believe to have arrived at an acceptable compromise between search time and loss reduction.

F. Real-Time Inference

Between the two sessions of the first experiment (training of the model and real-time usage) there is at least a difference of one day, which results in significant out-of-distribution data. This is very difficult to overcome as the common assumption for machine learning models is that the training and testing data are independent and identically distributed [38], [46], which clearly violates our knowledge about how the experiments are performed and more egregiously about the stochastisity of the EMG signal.

We trained the model using transfer learning, with data collected on the day of testing, for a period of 15 min to combat the possibility of faulty electrodes and to teach the model the new electrode grid position on the subject. Even with such measures the accuracy was not high enough to allow the user to believe that they had true volitional control.

The main problem with the model was its tendency to alternate between a resting state and the intended movement position, resulting in an inability to perform the desired movement with high accuracy. We hypothesize that both grid and arm position affect the predictive ability of the system. While the position of the grids can be adjusted through transfer learning techniques, correcting for the disparity in arm position between the testing and training phases is considerably more challenging. This is primarily due to the inherent difficulty in replicating the exact arm position since humans cannot consistently maintain the same position. To address this, we developed an algorithm that can correct the model's predictions in real-time, removing jitter and allowing all users to feel like they have control over the virtual hand. Our correction method differs from a standard low-pass filter because it does not introduce any delays, and it utilizes a memory of previous raw and filtered predictions to more accurately reconstruct the desired movement without slowing them down.

Fig. 3(a) illustrates the structure of the filter employed in our study. The effectiveness of the filter is demonstrated through a visual comparison between the raw and filtered predictions in Fig. 3(b) and Video 2. Fig 4 consists of two panels that highlight the filter's behavior when applied to both a near-perfect prediction (panel (a)), which represents an idealized scenario, and a noisy prediction (panel (b)) reflecting the realistic conditions encountered in our experiments.

This filtering behavior is achieved by following algorithm. For the first run (purple) the prediction P is stored into two queues, the prediction queue Q_P, which contains the last n predictions of the model, and the averaged prediction A queue Q_A, which contains the last n hand positions with all filtering applied to them. The output of the first cycle is the last element from Q_A.

For cycles 2 to n (red) the prediction is again saved in Q_P, after which it gets filtered with a basic smoothing filter (Eq. 2), that suppresses small changes (assuming they are random noise) and leaves big changes preserved (volitional changes).

$$A_{n-2} + \frac{\text{norm}\left(|P_{n-1} - A_{n-2}|\right)}{\text{weight}_{\text{smoothing filter}}} \cdot (P_{n-1} - A_{n-2}) \qquad (2)$$

The basic smoothing filter is a weighted addition between the current averaged prediction A_{n-1}, and the difference between the current prediction P_{n-1} and the previous averaged prediction A_{n-2}. The weight is defined by the division of the normalized difference we previously explained and an user settable parameter called "weight$_{\text{smoothing filter}}$". The combination resulting from the difference between P_{n-1} and A_{n-2} and the parameter value set decides the amount of smoothing and how much or less the new prediction resembles the actual prediction made by the model. This new prediction is saved into Q_A.

For all other cycles (blue), the prediction of the model is again stored in Q_P and having the basic smoothing filter (Eq. 2) described previously. In the next step, we build linear regressions for Q_P and Q_A. The regression models are used to predict one time step into the future (F_P and F_A) using data from Q_P and Q_A respectively. Then the average between the basic filtered prediction P_{n-1} and the regression prediction (F_P results in a while F_A in b) is built (Eq. 3).

$$a|b = \frac{F_{A|P} + P_{n-1} \cdot \text{weight}_{\text{prediction influence}}}{1 + \text{weight}_{\text{prediction influence}}} \qquad (3)$$

The low pass filtered prediction P_{n-1} can be weighted, so that the regression predictions influences the hand position more or less by setting the parameter "weight$_{\text{prediction influence}}$". Going one step back, we also take the resulting functions of the regression models and build the cross-correlation between them r. The reason why we used two regression models is that the Q_P contains all unfiltered information, with which we can detect changes or stops in the movement, however to the detriment of also containing jitter. Q_A contains only the smoothed positions. Using it, we can estimate how the next step would look like if we follow the movement trend with a smoothed jitter. Since we want to detect movement changes quickly while still smoothing out the jitter we need a combination of both regression models. This is performed in the following weighed average operation:

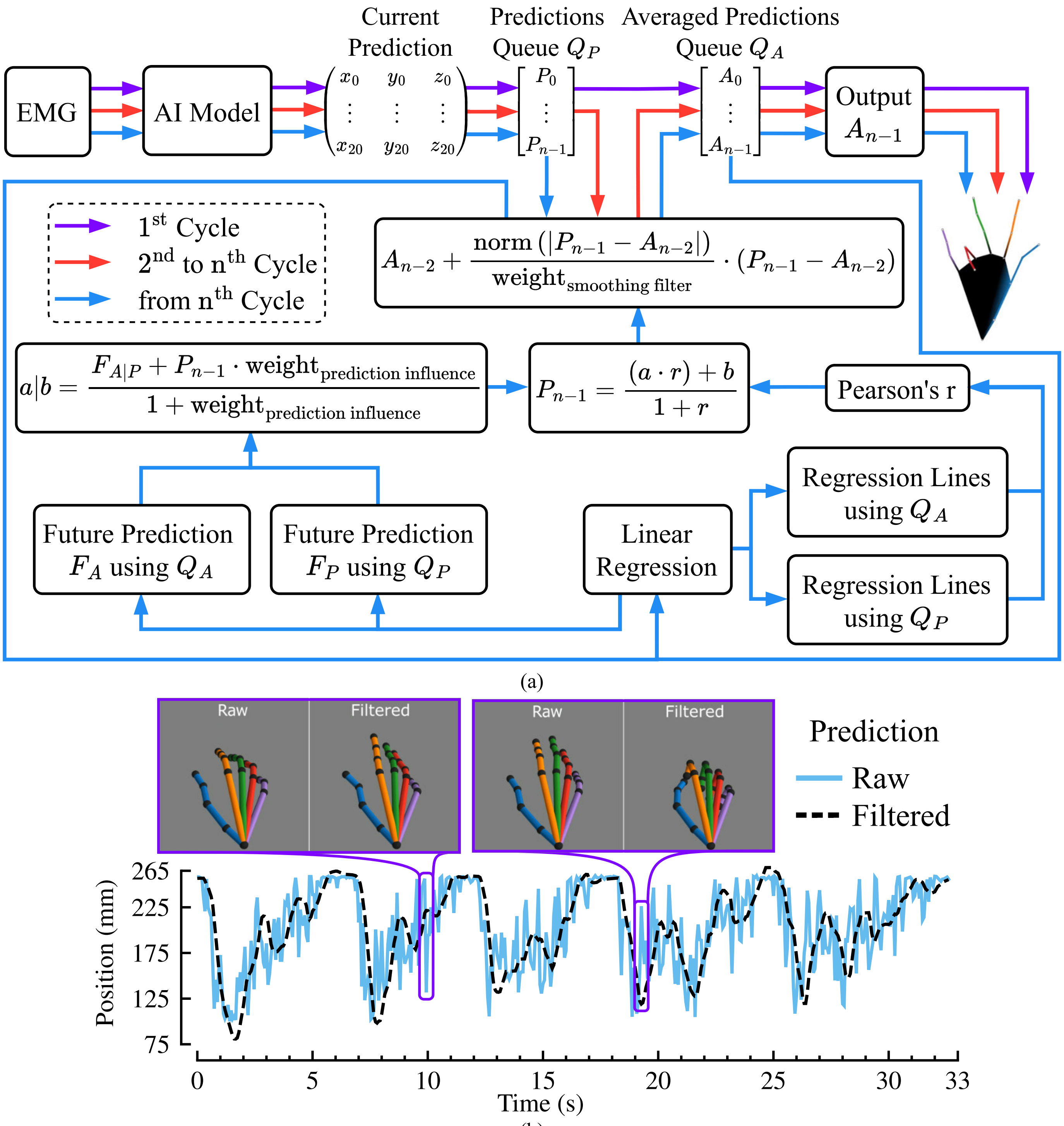
$$\text{P}_{\,n-1} = \frac{(a \cdot r) + b}{1 + r} \qquad (4)$$

If the correlation of the regressions r is low, the regression made from Q_P is weighted more heavily, because most likely a change in the movement happened that has to be taken into consideration otherwise we might negatively impact the feeling of volitional control. If necessary the prediction can be again filtered to suppress jittering even more by using the filter from Eq. 2.

Another challenge for real-time applications is latency. In addition to the necessary warm-up time of 93.75 ms (3 × 31.25 ms), which cannot be reduced as we need three small EMG segments (31.25 ms) to create the input of our model, we optimized the rest of our pipeline. We reduced latency by implementing our model in half-precision (float16) and using TorchScript, the PyTorch just-in-time compiler, to achieve a constant 32 predictions per second. Additionally, we wrote the correction prediction entirely in PyTorch. The code was executed on an NVIDIA RTX 3090 GPU during the real-time experiments.

G. Visualization

We used two libraries to visualize the 3D points that make up the hand joints. The first version used VisPy [47] because it ran on the GPU and allowed us to make a visualization



Current Prediction
Predictions Queue Q_P
Averaged Predictions Queue Q_A
EMG
AI Model
x_0 y_0 z_0
x_{20} y_{20} z_{20}
P_0
P_{n-1}
A_0
A_{n-1}
Output A_{n-1}
1st Cycle
2nd to nth Cycle
from nth Cycle
$A_{n-2} + \frac{\text{norm}(|P_{n-1} - A_{n-2}|)}{\text{weight}_{\text{smoothing filter}}} \cdot (P_{n-1} - A_{n-2})$
$a|b = \frac{F_{A|P} + P_{n-1} \cdot \text{weight}_{\text{prediction influence}}}{1 + \text{weight}_{\text{prediction influence}}}$
$P_{n-1} = \frac{(a \cdot r) + b}{1 + r}$
Pearson's r
Regression Lines using Q_A
Regression Lines using Q_P
Future Prediction F_A using Q_A
Future Prediction F_P using Q_P
Linear Regression
Raw
Filtered
Raw
Filtered
Prediction
Raw
Filtered
Position (mm)
265
225
175
125
75
0
5
10
15
20
25
30
33
Time (s)

(a)

(b)

Fig. 3. (a) An overview of the structure of the output smoothing filter. The algorithm has three levels of complexity based on the amount of time, and therefore on the available data history. We use 2 queues (Q_P for the prediction queue and Q_A for the averaged predictions) with a user-selectable length n. To denote the position and thus the time at which an element was added to the queue, we use zero indexing. This means that, for example, the current prediction is placed last in the queue and is therefore denoted as P_{n-1}. The oldest prediction stored is P_0. The very first cycle, represented by the purple arrows, forms the basis for all subsequent cycles. In the first cycle, the first prediction P is stored in both the predictions Q_P and the average predictions Q_A queues. The output is the unchanged AI prediction. Cycles 2 to n, where n is a hyperparameter selected by the user, are shown with red arrows. These cycles use Eq. 2 to smoothen the prediction P by applying an average filter over Q_P. The average value A is then stored in Q_A and then outputted. From the nth cycle (flow shown in blue) we consider to have enough data to begin our regressive prediction correction. Both Q_A and Q_P are used to fit 63 1-D lines (one for each coordinate for each joint) that we use to extract the movement trends. We also use the regression lines to get a prediction in the future and use the correlation between the movement trends to see how much to weight the future prediction based on actual AI output vs the filtered predictions. If a direction change were to occur we would favor the actual prediction (b in Eq. 4) where as if not we would favor the filtered prediction (a in Eq. 4). The output of this decision is then filtered by the running average and then outputted. (b) Plot showing a comparison of the raw and filtered trajectories of the middle finger tip predictions during grasping. The 3D frames seen are part of Video 2.

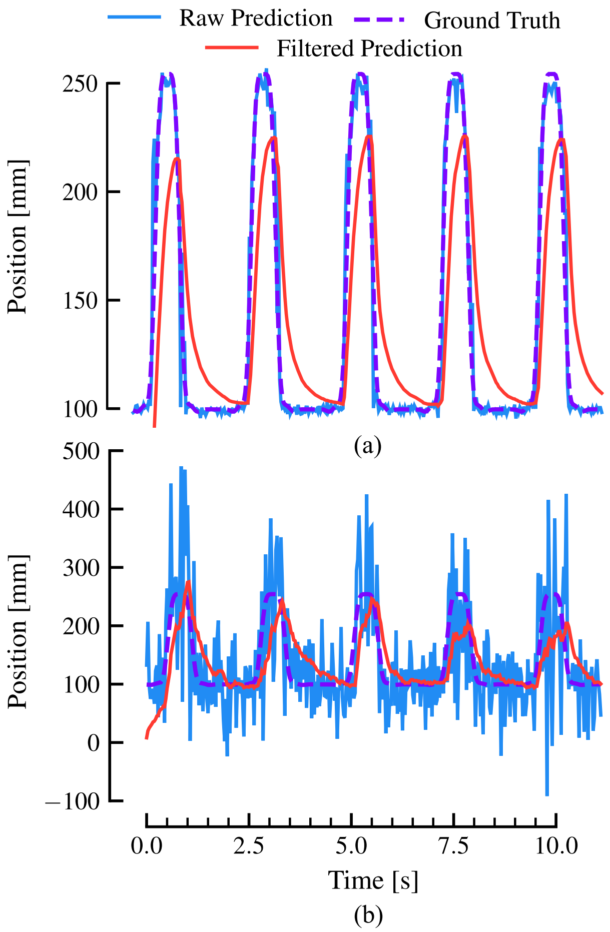


Fig. 4. Example of the real-time filter explained in Fig. 3. (a) With a perfect prediction, the filter suppresses the peaks. However, perfect predictions are very unlikely in reality. (b) With realistic predictive variability, the filter produces uniform output.

that was fast, intuitive, and most importantly not uncanny (Fig. 1, Fig. 3(a), and Video 2). To achieve the needed speed for 32 prediction per second we displayed the hand as 2D lines and shapes that gave the illusion of 3D. Although capable of displaying 3D tubes for the fingers VisPy needed (at the time of using it) to constantly recalculate each tube from its start and end points, meaning that each individual vertex that made up the surface needed to be recomputed which takes too long. We have used Mayavi [48] to create 3D visualizations of our data for offline analysis, as it allows us to translate the 3D tubes to new positions without the need to recompute them for each frame. This was not necessary for real-time rendering, but was useful for visualizing the results of the prediction correction (Fig. 3(b) and Video 2).

H. Statistical Analysis

An independent analysis of variance (ANOVA) was conducted to examine the task-wise averaged absolute and relative Euclidean distances in Fig. 5. The analysis was conducted in Python. Following that, post-hoc t-tests with Bonferroni correction were performed using the posthoc_ttest method from the scikit_posthocs library to determine statistical significances. Statistical significance was set at a p-value threshold of 0.05.

III. Results

We acquired sEMG data from 320 electrodes placed on the extrinsic forearm muscles (Fig. 2(b)) together with hand and wrist movements from 10 human subjects.

The first 9 subjects took part in the first experiment which consisted in following different movements using the trained models. During this experiment, we recorded both the guiding hand and the hand that was being controlled by the subject (Fig. 6(c)).

Subjects 1, 3, and 10 (new subject) participated in the second experiment, which consisted of reaching and holding various target positions in real time (Fig. 7(a)). In this experiment, the number of targets reached within 10 min, the type of target (Fig. 7(b) and (c)), and the time at which the target was reached and held for 2 s were recorded (Fig. 7(d)).

A. Movement-Following Experiment

The interface used in this experiment consists of the guiding and the predicted (using sEMG from the subject) hands. The colors of the guiding hand provide the subject with visual feedback based on the correlation coefficient (CC) between the ground truth and the prediction. If the CC is less than or equal to 0.75, the guiding hand is colored red. If the CC is greater than 0.75 but less than or equal to 0.90, the guiding hand is colored yellow, and if it is greater than 0.90, the guiding hand is colored green. Note that the CC is computed across all markers, therefore, this value takes into account also the static posture of the hand, since the neural network is continuously outputting all the three-dimensional points of the hand.

During the offline accuracy validation of the network (see below) we included both the R^2-Score of only the digits that were required to move as well as the Euclidean distance (the absolute error) across all the digits. The combination of these two metrics gives an index of the proportional 3D control of the hand that can be controlled by the subjects.

In Fig. 5 we display the mean Euclidean distance between the ground truth kinematics and three different prediction scenarios both in mm Fig. 5(a) and in percentage Fig. 5(b). The red line shows the maximum possible error achievable when the system predicts an all 0 vector. Due to the nature of the relative error calculation, which results in a constant value of 100% for the maximum error, we did not include the red line in Fig. 5(b). The purple lines shows the error when the output is a static stretched out hand, which simulates failure to detect any movement intent by predicting resting position. The AI prediction, averaged across the subjects, is shown in blue.

Using the recorded videos we have visually inspected if movement intent resembling the intended movement could be detected. For example, if the flexion of the ring finger is detected half way between extension and flexion, then it is considered detected, while a flexion of the middle finger instead of the ring finger is not considered as correct. The movement intent detection was performed visually and the results were then combined with the two metrics (Euclidean distance for all the markers of the hand and the R^2-Score across the fingers that were required to move with the guiding hand signal). The

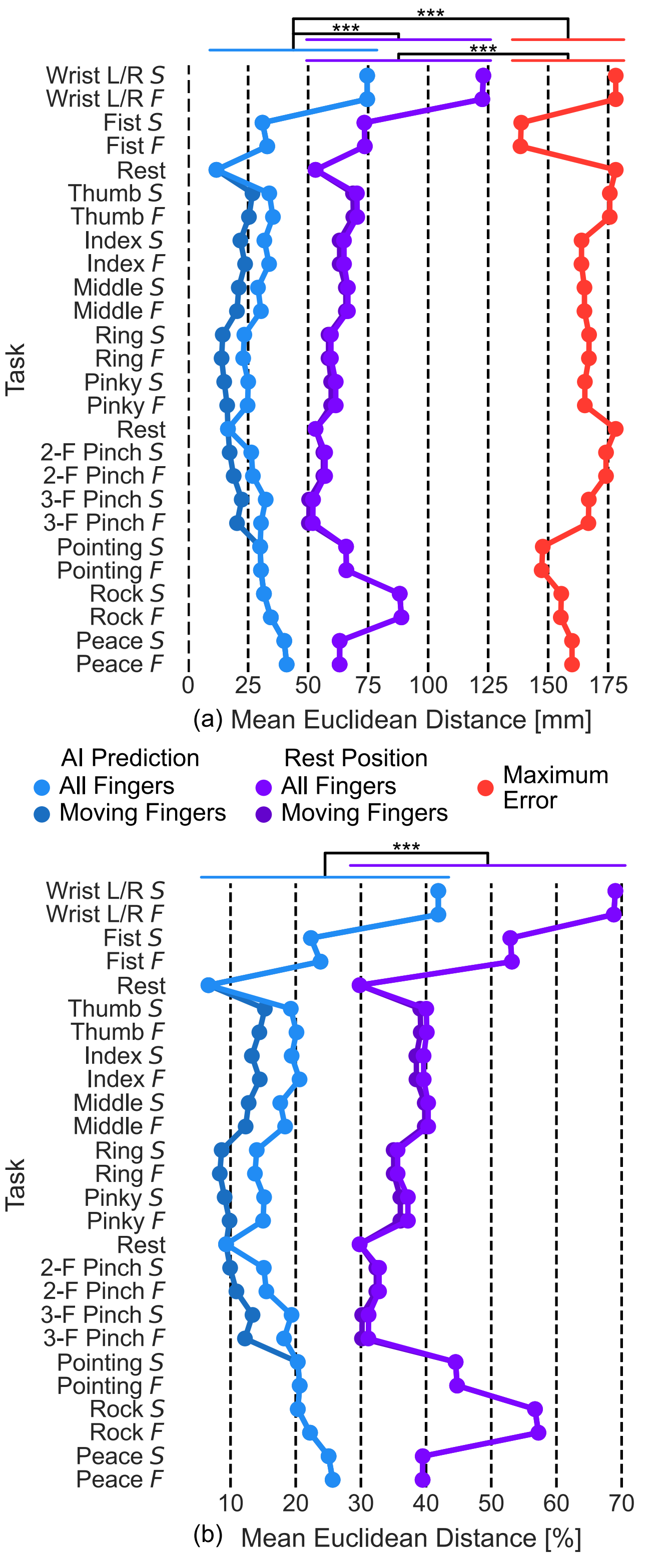


Fig. 5. Mean Euclidean distance between ground truth data and predictions. (a) The absolute error in mm between the AI prediction, a steady prediction representing the rest state, and a prediction where all values are set to zero. The averaging of the rest state and AI predictions is displayed for both all fingers and only the moving fingers. (b) The relative error in percentage. The normalization is performed using the maximum error from (a) task-wise. The statistical significance was assessed by conducting post-hoc t-tests with Bonferroni correction following an independent analysis of variance (ANOVA). *** = $p < 0.001$.

results of this analysis are displayed in Fig. 6(a) as a binary heatmap. Overlaid on top is the interquartile range (IQR) of the Euclidean distance in mm averaged over all joints and all time points. The mean Euclidean distance across all subjects is 36 ± 34 mm. The median distance is 21 mm, and the IQR spans from 9.8 to 42.8 mm.

Fig. 6(b) shows the R^2-Scores (mean across subjects: 0.65) of the movements. The R^2-Scores between the ground truth and the prediction was calculated by averaging over all finger tips and the first joints from the finger tips (distal interphalangeals). Both the ground truth and the prediction were first filtered with a 0.75 Hz low-pass filter before the averaging was performed. By using this method, we were able to compare the movement intent between the ground truth and the prediction, regardless of whether the subjects were perfectly synchronized with the guiding hand or not. Two of the videos used for this analysis (Fig. 6(b)) showing subject 2 and 7 are available as Video 1 and 3.

B. Target Reaching Experiment

Three participants were recruited to perform a series of target-reaching tasks using our user-in-the-loop system within a 10 min time constraint. The objective of the experiment was twofold: to assess the feasibility of reaching all target positions and to measure the speed at which they could be achieved.

The interface displayed the predicted hand together with colored targets (dots) that matched the color of the individual fingers. The arrangement of the colored targets changes to depict the desired hand positions (Fig. 7(a)). A target is considered reached if the averaged distance between the colored targets and their respective fingers is below 20 mm for 2 s. To further aid the subjects we displayed a short target description above their virtual hand (e.g. “fist” for closing the fist) and changed the color of the text from red to green if the error is below 20 mm (Fig. 7(a)). If the target is achieved or if 10 s pass without reaching it, the next target is chosen in a way that ensures consecutive targets are never identical.

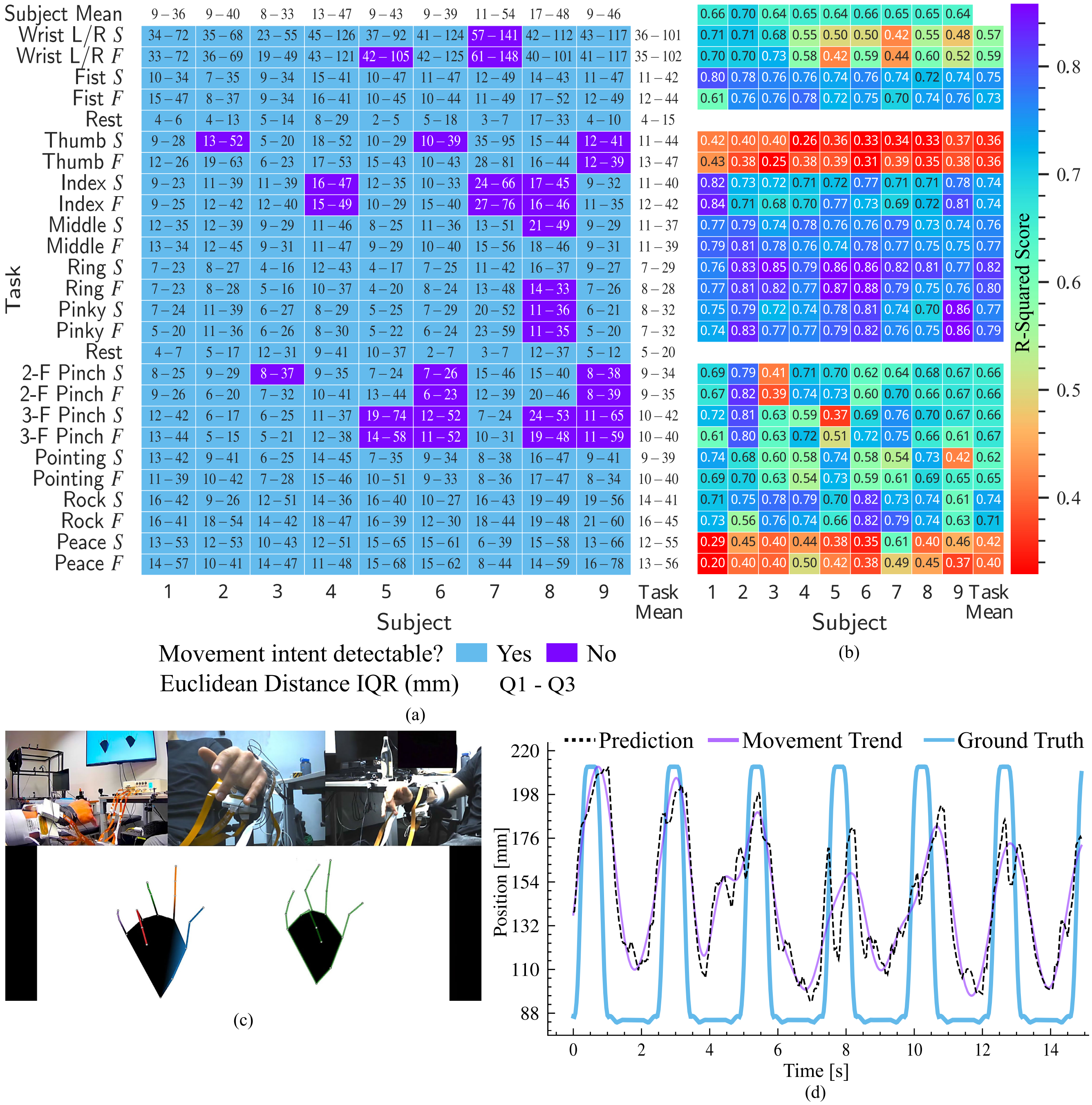
This target selection rule causes the selection to no longer be uniformly distributed. The recorded distribution of the tasks can be seen in Fig. 7(b).

In Fig. 7(c), we present the number of reached and not reached targets for each subject. The total count of reached targets is shown on the right, along with the calculated completion rate. The 3 subjects achieved completion rates of 90%, 82%, and 83%, respectively, resulting in a mean of 85%.

Fig. 7(d) displays the duration taken to reach the targets. The reaching moment is defined as the point when the subject was first below the 20 mm distance requirement of the target, provided that position was sustained for 2 s. The mean time to reach was 2.23 ± 1.83 s, while the median is 2.06 s (IQR 0.27–3.66 s).

C. Latency

The latency of our system (Fig. 8) has been calculated in simulated real-time using 100000 inputs. We then confirmed these results during the actual real-time experiments. After the initial warm up of 93.75 ms our system needs to wait for one small EMG segment (31.25 ms) to be acquired by the recording system. Afterwards both the model inference and the



Task	1	2	3	4	5	6	7	8	9	Task Mean
Subject Mean	9 – 36	9 – 40	8 – 33	13 – 47	9 – 43	9 – 39	11 – 54	17 – 48	9 – 46	
Wrist L/R *S*	34 – 72	35 – 68	23 – 55	45 – 126	37 – 92	41 – 124	57 – 141	42 – 112	43 – 117	36 – 101
Wrist L/R *F*	33 – 72	36 – 69	19 – 49	43 – 121	42 – 105	42 – 125	61 – 148	40 – 101	41 – 117	35 – 102
Fist *S*	10 – 34	7 – 35	9 – 34	15 – 41	10 – 47	11 – 47	12 – 49	14 – 43	11 – 47	11 – 42
Fist *F*	15 – 47	8 – 37	9 – 34	16 – 41	10 – 45	10 – 44	11 – 49	17 – 52	12 – 49	12 – 44
Rest	4 – 6	4 – 13	5 – 14	8 – 29	2 – 5	5 – 18	3 – 7	17 – 33	4 – 10	4 – 15
Thumb *S*	9 – 28	13 – 52	5 – 20	18 – 52	10 – 29	10 – 39	35 – 95	15 – 44	12 – 41	11 – 44
Thumb *F*	12 – 26	19 – 63	6 – 23	17 – 53	15 – 43	10 – 43	28 – 81	16 – 44	12 – 39	13 – 47
Index *S*	9 – 23	11 – 39	11 – 39	16 – 47	12 – 35	10 – 33	24 – 66	17 – 45	9 – 32	11 – 40
Index *F*	9 – 25	12 – 42	12 – 40	15 – 49	10 – 29	15 – 40	27 – 76	16 – 46	11 – 35	12 – 42
Middle *S*	12 – 35	12 – 39	9 – 29	11 – 46	8 – 25	11 – 36	13 – 51	21 – 49	9 – 29	11 – 37
Middle *F*	13 – 34	12 – 45	9 – 31	11 – 47	9 – 29	10 – 40	15 – 56	18 – 46	9 – 31	11 – 39
Ring *S*	7 – 23	8 – 27	4 – 16	12 – 43	4 – 17	7 – 25	11 – 42	16 – 37	9 – 27	7 – 29
Ring *F*	7 – 23	8 – 28	5 – 16	10 – 37	4 – 20	8 – 24	13 – 48	14 – 33	7 – 26	8 – 28
Pinky *S*	7 – 24	11 – 39	6 – 27	8 – 29	5 – 25	7 – 29	20 – 52	11 – 36	6 – 21	8 – 32
Pinky *F*	5 – 20	11 – 36	6 – 26	8 – 30	5 – 22	6 – 24	23 – 59	11 – 35	5 – 20	7 – 32
Rest	4 – 7	5 – 17	12 – 31	9 – 41	10 – 37	2 – 7	3 – 7	12 – 37	5 – 12	5 – 20
2-F Pinch *S*	8 – 25	9 – 29	8 – 37	9 – 35	7 – 24	7 – 26	15 – 46	15 – 40	8 – 38	9 – 34
2-F Pinch *F*	9 – 26	6 – 20	7 – 32	10 – 41	13 – 44	6 – 23	12 – 39	20 – 46	8 – 39	9 – 35
3-F Pinch *S*	12 – 42	6 – 17	6 – 25	11 – 37	19 – 74	12 – 52	7 – 24	24 – 53	11 – 65	10 – 42
3-F Pinch *F*	13 – 44	5 – 15	5 – 21	12 – 38	14 – 58	11 – 52	10 – 31	19 – 48	11 – 59	10 – 40
Pointing *S*	13 – 42	9 – 41	6 – 25	14 – 45	7 – 35	9 – 34	8 – 38	16 – 47	9 – 41	9 – 39
Pointing *F*	11 – 39	10 – 42	7 – 28	15 – 46	10 – 51	9 – 33	8 – 36	17 – 47	8 – 34	10 – 40
Rock *S*	16 – 42	9 – 26	12 – 51	14 – 36	16 – 40	10 – 27	16 – 43	19 – 49	19 – 56	14 – 41
Rock *F*	16 – 41	18 – 54	14 – 42	18 – 47	16 – 39	12 – 30	18 – 44	19 – 48	21 – 60	16 – 45
Peace *S*	13 – 53	12 – 53	10 – 43	12 – 51	15 – 65	15 – 61	6 – 39	15 – 58	13 – 66	12 – 55
Peace *F*	14 – 57	10 – 41	14 – 47	11 – 48	15 – 68	15 – 62	8 – 44	14 – 59	16 – 78	13 – 56

Movement intent detectable? Yes No

Euclidean Distance IQR (mm) Q1 - Q3

(a)

	1	2	3	4	5	6	7	8	9	Task Mean
	0.66	0.70	0.64	0.65	0.64	0.66	0.65	0.65	0.64	
	0.71	0.71	0.68	0.55	0.50	0.50	0.42	0.55	0.48	0.57
	0.70	0.70	0.73	0.58	0.42	0.59	0.44	0.60	0.52	0.59
	0.80	0.78	0.76	0.76	0.74	0.76	0.74	0.72	0.74	0.75
	0.61	0.76	0.76	0.78	0.72	0.75	0.70	0.74	0.76	0.73
	0.42	0.40	0.40	0.26	0.36	0.33	0.34	0.33	0.37	0.36
	0.43	0.38	0.25	0.38	0.39	0.31	0.39	0.35	0.38	0.36
	0.82	0.73	0.72	0.71	0.72	0.77	0.71	0.71	0.78	0.74
	0.84	0.71	0.68	0.70	0.77	0.73	0.69	0.72	0.81	0.74
	0.77	0.79	0.74	0.78	0.76	0.76	0.79	0.73	0.74	0.76
	0.79	0.81	0.78	0.76	0.74	0.78	0.77	0.75	0.75	0.77
	0.76	0.83	0.85	0.79	0.86	0.86	0.82	0.81	0.77	0.82
	0.77	0.81	0.82	0.77	0.87	0.88	0.79	0.75	0.76	0.80
	0.75	0.79	0.72	0.74	0.78	0.81	0.74	0.70	0.86	0.77
	0.74	0.83	0.77	0.77	0.79	0.82	0.76	0.75	0.86	0.79
	0.69	0.79	0.41	0.71	0.70	0.62	0.64	0.68	0.67	0.66
	0.67	0.82	0.39	0.74	0.73	0.60	0.70	0.66	0.67	0.66
	0.72	0.81	0.63	0.59	0.37	0.69	0.76	0.70	0.67	0.66
	0.61	0.80	0.63	0.72	0.51	0.72	0.75	0.66	0.61	0.67
	0.74	0.68	0.60	0.58	0.74	0.58	0.54	0.73	0.42	0.62
	0.69	0.70	0.63	0.54	0.73	0.59	0.61	0.69	0.65	0.65
	0.71	0.75	0.78	0.79	0.70	0.82	0.73	0.74	0.61	0.74
	0.73	0.56	0.76	0.74	0.66	0.82	0.79	0.74	0.63	0.71
	0.29	0.45	0.40	0.44	0.38	0.35	0.61	0.40	0.46	0.42
	0.20	0.40	0.40	0.50	0.42	0.38	0.49	0.45	0.37	0.40

(b)

(c)

(d)

Fig. 6. (a) The binary heatmap shows if the movement intent can be visually detected from recordings done during the experiments (see (c), and Video 1 and 3). Overlaid on top is the interquartile range (IQR) of the mean Euclidean distance in mm for that respective task/subject pair. The task suffix *S* indicates a slow movement, while *F* indicates a fast movement. (b) The heatmap shows the R^2-Score for each movement averaged over all finger tips. Rest position is not displayed as this is a static pose for which our system outputs a very similar and slightly noisy signal. Displaying it will results in negative R^2-Scores even though from (a) we can see that the errors are negligible. (c) One frame from Video 1 used to determine if the movement intent is detectable. (d) The trajectory comparison of the middle finger tip. The movement trend is the 1 Hz low-pass prediction and shows that the general movement is proportionally reproduced with relatively good accuracy.

corrections are negligible. The model inference takes about 6.25 ± 10.54 ms with a 99% confidence interval (CI) lying between 6.18 and 6.41 ms. The prediction correction takes 1.17 ± 0.53 ms with a 99% CI between 1.162 and 1.169 ms. In total the latency of our system is about 38.7 ms. The prediction process, including model inference and correction, takes less than 31.25 ms, allowing us to sustain a constant rate of 32 predictions per second.

IV. Discussion

We trained a deep learning model based on the subject-specific EMG activity to predict hand kinematics. This model was then utilized in two separate experiments conducted over a period of several months. In our study, we have successfully demonstrated the real-time predictive capabilities of our deep learning model in relation to the kinematics of 12 human hand movements. Across various subjects, our

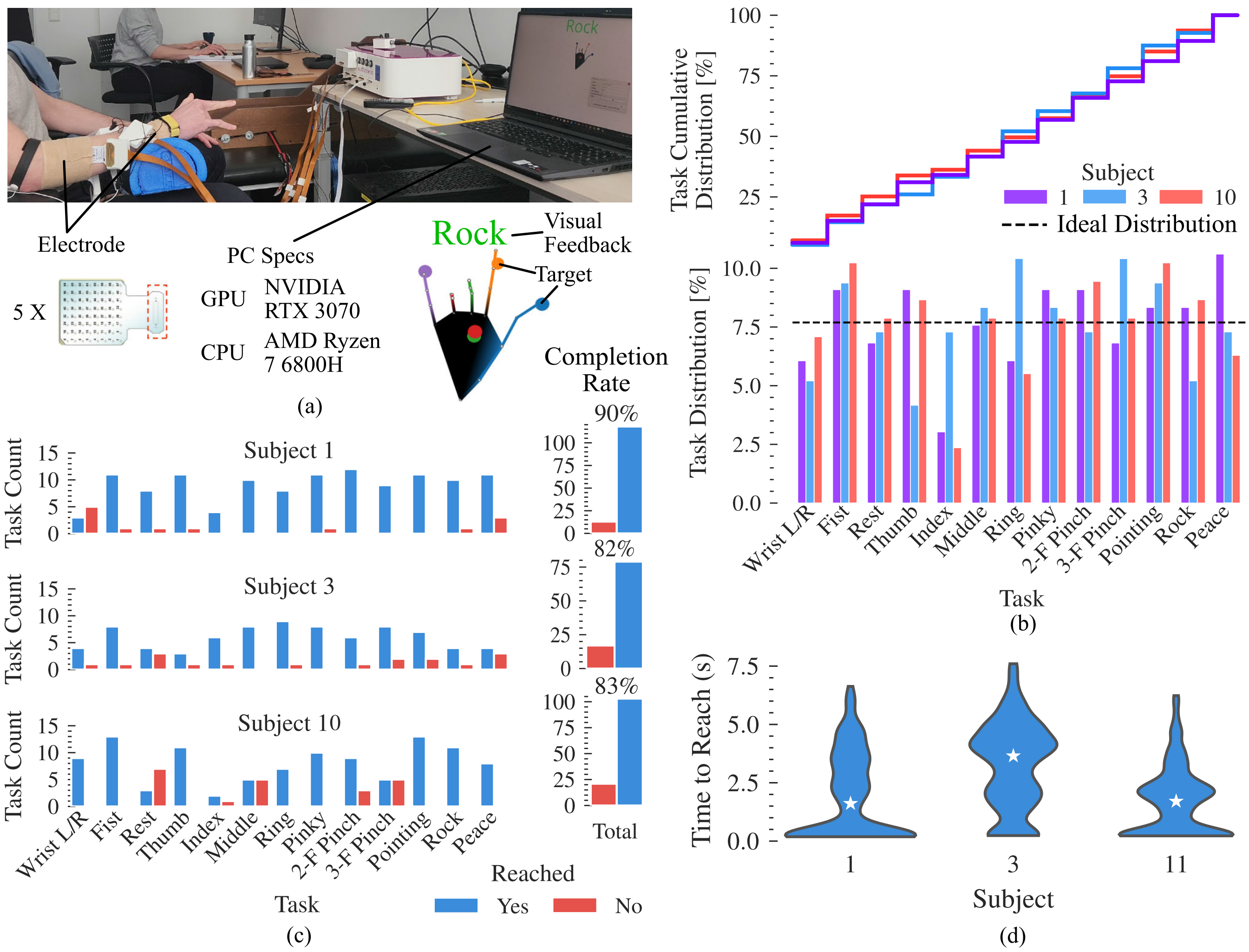


Fig. 7. (a) Target reaching experimental setup. Three subjects (1, 3, and 10) had 5 electrode grids placed on their forearm. A model was trained for each subject from zero using 20 s recordings of 13 different movements (see x-axis of (b)). After the model training, the subjects were given 5 min to familiarize themselves with the closed-loop interface. The interface displayed a hand with differently colored fingers and dots of the same color as the fingers for each movement target. The interface also indicated which movement should be executed and whether it was reached using red (not reached) and green (reached) text above the hand. The subjects were then asked to reach as many targets they could in 10 min. A target is considered reached if a distance smaller than 20 mm is held for 2 s. After 10 s of not reaching a target the target would randomly select the next one. (b) Distribution of the targets shown. The ideal distribution would be uniform but due to the experimental necessity to not select the same target twice in a row the actual distribution differed. (c) Amount of targets reached and not reached. The total number differs between subjects as the faster the targets are reached the more are shown withing the 10 min. (d) Violin plots showing time to reach. The moment of reaching is considered to be the starting moment when the subject is below 20 mm after the target has been successfully held for 2 s. We display the median for each subject with a star shape (1.61, 3.65, and 1.71 s).

model achieved a mean Euclidean error of 3.6 cm (Fig. 6(a)) and exhibited a completion rate of 85% when tasked with reaching different targets (Fig. 7(c)).

In our first experiment, we conducted a study involving 9 subjects who were instructed to perform various dynamic movements and hand gestures while recording sEMG signals and kinematic data. The movements included wrist adduction and abduction, flexion and extension of each digit, fist grasping (closing and opening), as well as two- and three-finger pinches. Additionally, three hand gestures (pointing, rock sign, and peace sign) were performed. Subject 1 was instructed to perform the kinematics prior to the experiment, and their recorded kinematic data were subsequently used as a reference for all other participants in the study. By utilizing the kinematics from subject 1 as a reference for all participants, we were able to mitigate the need for individual data collection. This approach not only streamlines the process but also opens up possibilities for testing individuals with paralysis or amputations, as their hand movements cannot be directly recorded due to it either not moving or being absent. By utilizing techniques such as resampling, we have the flexibility to modify the speed of recorded movements. This adaptability opens up possibilities for individuals with impairments to effectively track and replicate movements of a virtual hand. In our previous study [9], we demonstrated the feasibility of this concept by employing a comparatively straightforward machine learning approach, allowing patients with spinal cord injuries to regain limited control over a few DoFs. This was only possible because we could leverage the kinematics of a healthy individual as the ground truth that the spinal cord injury patient then saw and

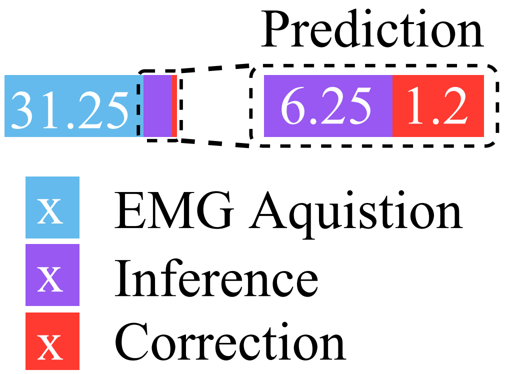


Fig. 8. Timeline explanation showing breakdown of the latency. All units shown are in ms. After the initial warm up of 93.75 ms, the input queue must be shifted one EMG segment over. This results in a 31.25 ms delay (waiting signal), as the new EMG segment must be acquired by our system. Afterwards we run the model on the new input and apply the prediction correction to remove any unwanted artifacts in the movements. The latency for inference and the prediction algorithm have been determined in a simulated real-time scenario by using 100000 long segments. The real-time experiments showed similar performances. The exact latency for the inference is 6.25 ± 10.54 ms with a 99% confidence interval (CI) lying between 6.18 and 6.41 ms. The prediction correction takes 1.17 ± 0.53 ms with a 99% CI between 1.162 and 1.169 ms.

attempted to follow. However, it is important to acknowledge that using subject 1's kinematics as a reference may introduce challenges in decoding, as not all participants exhibit the same finger enslavings (movement of additional fingers along with the intended finger) or perform the movements precisely at the requested timing. This hypothesis should be explicitly evaluated and validated in future research.

Using this dataset, we trained subject-specific models that were subsequently tested in real-time scenarios. However, due to the significant variability observed in the raw unfiltered monopolar sEMG signals across different recordings and hand postures, we employed transfer learning prior to predicting hand kinematics in real-time. Transfer learning involves rapidly training a pre-existing model on new data (approximately 15 min in our case), taking into account data drifts that may occur due to factors such as variations in the placement of the electrode grids on the skin (Fig. 2(b)).

Following the transfer learning process, the subjects were required to perform different movements in real-time by following a guiding hand. The real-time predictions were recorded and analyzed in Fig. 6, Video 1 and 3.

Although our system achieved successful decoding in most tasks, there were certain instances where it did not perform as expected. For example, subjects 6 and 9, were unable to perform pinching movements, but they could effectively close and open the virtual fist. Subject 4, 7, and 8 experienced limited responsiveness in their index finger, yet it remained controllable during combined tasks such as fist closure and opening. We speculate that these errors could be attributed to potential overfitting of the models for these specific subjects, primarily on hand gestures, and the failure to adequately account for EMG variations between the original recording and real-time sessions during the brief transfer learning period.

It is also important to note that the kinematics used to train the model were recorded from subject 1, who is the only subject able to correctly execute all movements. We speculate that the AI might have learned some invariant neural dynamics underlying the movement of the hand. Although this study cannot demonstrate that this is the case, our previous purely offline systems [29], [30] seem to also be in support of this idea as they achieve good results on models trained from subject-dependent kinematics.

The Euclidean distances presented in Fig. 6(a) are primarily influenced by the accuracy of our models in decoding the intended movements. However, they are also subject to additional factors such as the synchronization between the subject and the guiding hand, the level of fatigue experienced by the subject, and the ability to reproduce smooth flexion-extension phases. We chose it as the metric in our study because all subjects followed the same kinematic patterns, enabling comparability of results across individuals. By stabilizing the hand position using the wrist as the origin in Cartesian coordinates, it becomes easier to intuitively comprehend errors in terms of millimeters or centimeters relative to a target, as opposed to using angular measurements. This choice is motivated by the observation that even a small angular deviation at the metacarpophalangeal joint (base of the finger) can result in a significantly larger Euclidean error compared to the distal interphalangeal joint (at the fingertip). The presence of a discontinuity in error contribution, as observed our previous study (Sîmpetru et al. [30]), not only poses challenges for generalization when utilizing angular coordinates but also has implications for accurately assessing the prediction capabilities of the system. This discontinuity can potentially lead the reader to misinterpret or underestimate the system's predictive performance.

The thumb and peace sign tasks exhibited lower R^2-Scores compared to the other tasks. The lower score observed in the thumb task can be attributed to the anatomical placement of the thumb muscles. The majority of muscles responsible for controlling the thumb are located within the intrinsic hand region. The peace sign task involves a slight flexion of the wrist, which could potentially introduce movement artifacts in the wrist grids. These artifacts may contribute to a noisier signal, leading to a less smooth and less accurate decoding compared to the other tasks.

In the movement-following experiment illustrated in Fig. 6, our system demonstrated an average Euclidean distance of approximately 3.6 cm per subject when tracking sinusoidal movements (Fig. 6(a)). The corresponding R^2-scores (Fig. 6(b)) and the provided example prediction (Fig. 6(d)) indicate that the system successfully follows the intended movements, with the primary source of error being the delayed execution of flexion/extension motions.

To assess the impact of this level of error on the practical usability of our system, we conducted an additional experiment involving three subjects. This experiment was designed to test the user-in-the-loop experience and required the subjects to reach as many target positions as possible within 10 min (Fig. 7). The targets are depicted as colored dots that matched the color of the virtual hand's fingers. A target position was considered reached if the mean distance between the target dots and their respective fingers was smaller then 20 mm for a duration of 2 s. In comparison to other studies in the literature, such as the work by Nowak et al. [8] with a hold time of 0.3 s, our choice of a 2 s hold time is significantly longer. We deliberately opted for this longer duration as it aligns more closely with real-life scenarios where the objective is often to maintain a grasp on an object for extended periods rather than solely reaching towards it (e.g., holding an umbrella).

The results in Fig. 7(c) indicate that all targets were reachable by the users, although individual subjects exhibited varying patterns of failure. This could be due to the arm position the subject had at the time which causes data drifts that the model can not compensate. A problem that could be mitigated during training by providing different arm positions for the same movement. Furthermore, the reach time (Fig. 7(d)) is also influenced by the aforementioned problem of data drift resulting from inconsistent arm positions. This hypothesis is supported by the observation that two subjects (1 and 10) exhibited a higher number of reaching times falling within the typical human reaction speed (as indicated by the base of the violin plot). It is plausible to suggest that these instances corresponded to situations where the arm position closely matched the configuration during training, enabling the system to make more accurate predictions.

While not perfect, our system is able to decode a greater number of distinct movements than previously published solutions [3], [4], [5], [6], [7], [8], [28], at a rate of 32 predictions per second, providing a sense of embodiment and volitional control. Our system not only demonstrates the ability to decode a larger set of proportional movements but also maintains a high completion rate, as depicted in (Fig. 7(c)), in comparison to other real-time studies that focus on a narrower range of movements [7], [8]. The proposed system may lay the foundation for an EMG processing pipeline that can take leverage of the full EMG bandwidth, without imposing any a priori constraint but by learning how the brain controls the human hand through feedforward processing of the monopolar, raw EMG signals.

In a prior study [30], we conducted an investigation using nonlinear factorization methods to analyze the features acquired by the neural network. Our findings revealed that the model successfully distinguished different hand movements, displaying distinct clusters for each individual finger and discernible separation between flexion and extension movements of the same finger. This is of high importance because this demonstrates that the proposed system can exploit the nonlinear associations within the EMG signals unlike other approaches [7], [8], [9] that reduce the EMG signal to linear mappings in order to achieve better robustness. From a physiological perspective this result is not so surprising since during dynamic movements there are several nonlinear components in the EMG signals both at the neural and muscular level which are likely only depicted by the full frequency band of the EMG (20-500 Hz, [25]).

Although the decomposition of the high-density EMG represents the most adequate solution since it mimics how the central nervous system encodes muscle forces [9], [49], [50], there are a large number of limitations in decoding a significant number of motor units during dynamic hand movement in real-time (see our previous conference paper [32] and tutorial article [27]) due to the high nonlinearities in the action potential shapes that are distorted by the contracting muscles [25]. Here we argue that machine learning, and more specifically deep learning, may represent the future for interfacing human movement with machines using the surface EMG signals.

V. Conclusion

In this study we acquired sEMG data from 320 electrodes placed on the extrinsic hand muscles (Fig. 2(b)) together with hand movements from 10 human subjects. The subject had to perform dynamic movements (adducting and abducting the wrist, flexing and extending each digit, closing and opening the fist to grasp, and 2- and 3- finger pinches) as well as 3 hand gestures (pointing, rock sign, and peace sign).

We used the recorded data to train subject-specific models that are able to decode most performed movements in real-time. To achieve this, we conducted a short (15 min) transfer learning session before testing the system in real-time, and developed a prediction correction algorithm to remove artifacts and smooth the output. We also asked 3 of the 10 subjects to attempt to reach as many targets as possible in a three-dimensional Cartesian space in real-time in order to test the user-in-the-loop experience and robustness of the proposed method.

Our system outperforms previously published systems in both number of movements that can be performed and reliability. Furthermore, the proposed system is capable of proportionally controlling a greater number of movements, and similar completion rates despite the fact that previous solutions focus on signifcantly smaller sets of movements. The current work also highlights future challenges that deep learning myocontrol algorithms should overcome such as overfitting and dealing with the complexity of EMG changes with specific movement patterns. The proposed solution presented in this work may serve as a foundation for the potential development of reliable real-time movement decoders leveraging surface high-density EMG activity.

Acknowledgment

The authors would like to thank the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR at FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU). The hardware is funded by the German Research Foundation (DFG). They also would like to thank Vlad Cnejevici for his support during the recording of the second experiment.

References

[1] S. A. Stuttaford, S. S. G. Dupan, K. Nazarpour, and M. Dyson, “Training prosthesis control in the lab and the home,” in *Myoelectric Controls and Upper Limb Prosthetics Symposium*. Fredericton, NB, Canada: Institute of Biomedical Engineering, University of New Brunswick, Sep. 2022, pp. 90–93. [Online]. Available: https://www.unb.ca/ibme/_assets/documents/mec22proceedings2.pdf

[2] I. Vujaklija and D. Farina, “Prosthetics and innovation,” in *Blast Injury Science and Engineering: A Guide for Clinicians and Researchers*. Berlin, Germany: Springer Nature, 2023, pp. 421–437.

[3] N. Jiang, J. L. Vest-Nielsen, S. Muceli, and D. Farina, "EMG-based simultaneous and proportional estimation of wrist/hand kinematics in uni-lateral trans-radial amputees," *J. NeuroEngineering Rehabil.*, vol. 9, no. 1, p. 42, Dec. 2012.

[4] S. Amsuess, P. Goebel, B. Graimann, and D. Farina, "A multi-class proportional myocontrol algorithm for upper limb prosthesis control: Validation in real-life scenarios on amputees," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 23, no. 5, pp. 827–836, Sep. 2015.

[5] K. Z. Zhuang et al., "Shared human–robot proportional control of a dexterous myoelectric prosthesis," *Nature Mach. Intell.*, vol. 1, no. 9, pp. 400–411, Sep. 2019.

[6] D. Yang and H. Liu, "An EMG-based deep learning approach for multi-DOF wrist movement decoding," *IEEE Trans. Ind. Electron.*, vol. 69, no. 7, pp. 7099–7108, Jul. 2022.

[7] C. Chen, Y. Yu, X. Sheng, D. Farina, and X. Zhu, "Simultaneous and proportional control of wrist and hand movements by decoding motor unit discharges in real time," *J. Neural Eng.*, vol. 18, no. 5, Oct. 2021, Art. no. 056010.

[8] M. Nowak, I. Vujaklija, A. Sturma, C. Castellini, and D. Farina, "Simultaneous and proportional real-time myocontrol of up to three degrees of freedom of the wrist and hand," *IEEE Trans. Biomed. Eng.*, vol. 70, no. 2, pp. 459–469, Feb. 2023.

[9] D. Souza de Oliveira et al., "You will grasp again: A direct spinal cord/computer interface with the spared motor neurons restores the dexterous control of the paralyzed hand after chronic spinal cord injury," Sep. 2022, doi: 10.1101/2022.09.09.22279611.

[10] B. Waryck, "Comparison of two myoelectric multi-articulating prosthetic hands," in *Proc. MEC Conf.*, 2011, pp. 1–14.

[11] M. G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, and A. Bicchi, "Adaptive synergies for the design and control of the pisa/IIT SoftHand," *Int. J. Robot. Res.*, vol. 33, no. 5, pp. 768–782, Apr. 2014.

[12] Z. Xu and E. Todorov, "Design of a highly biomimetic anthropomorphic robotic hand towards artificial limb regeneration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 3485–3492.

[13] M. Controzzi, F. Clemente, D. Barone, A. Ghionzoli, and C. Cipriani, "The SSSA-MyHand: A dexterous lightweight myoelectric hand prosthesis," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 5, pp. 459–468, May 2017.

[14] J. Fajardo, V. Ferman, D. Cardona, G. Maldonado, A. Lemus, and E. Rohmer, "Galileo hand: An anthropomorphic and affordable upper-limb prosthesis," *IEEE Access*, vol. 8, pp. 81365–81377, 2020.

[15] J. M. Hahne, M. A. Wilke, M. Koppe, D. Farina, and A. F. Schilling, "Longitudinal case study of regression-based hand prosthesis control in daily life," *Frontiers Neurosci.*, vol. 14, p. 600, Jun. 2020.

[16] M. Laffranchi et al., "The hannes hand prosthesis replicates the key biological properties of the human hand," *Sci. Robot.*, vol. 5, no. 46, Sep. 2020, Art. no. eabb0467.

[17] M. N. Castro and S. Dosen, "Continuous semi-autonomous prosthesis control using a depth sensor on the hand," *Frontiers Neurorobotics*, vol. 16, Mar. 2022, Art. no. 814973.

[18] A. Bayrak and E. Bekiroglu, "Bionic hand: A brief review," *J. Bionic Memory*, vol. 2, pp. 37–43, Mar. 2022.

[19] A. Marinelli et al., "Active upper limb prostheses: A review on current state and upcoming breakthroughs," *Prog. Biomed. Eng.*, vol. 5, no. 1, p. 012001, Jan. 2023, doi: 10.1088/2516-1091/acac57.

[20] E. Biddiss and T. Chau, "Upper-limb prosthetics: Critical factors in device abandonment," *Amer. J. Phys. Med. Rehabil.*, vol. 86, no. 12, pp. 977–987, Dec. 2007.

[21] S. Salminger et al., "Current rates of prosthetic usage in upper-limb amputees–have innovations had an impact on device acceptance?" *Disability Rehabil.*, vol. 44, no. 14, pp. 3708–3713, Jul. 2022, doi: 10.1080/09638288.2020.1866684.

[22] E. R. Kandel, *Principles of Neural Science*, 4th ed. New York, NY, USA: McGraw-Hill, 2000.

[23] J. V. Basmajian and C. J. De Luca, *Muscles Alive: Their Functions Revealed by Electromyography*, 5th ed. Baltimore, MD, USA: Williams & Wilkins, 1985.

[24] R. Merletti and P. Parker, *Electromyography: Physiology, Engineering, and Noninvasive Applications* (IEEE Press Series in Biomedical Engineering). Hoboken, NJ, USA: IEEE/John Wiley, 2004.

[25] R. Merletti and D. Farina, *Surface Electromyography: Physiology Engineering and Applications* (IEEE Press Series in Biomedical Engineering). Piscataway, NJ, USA: IEEE Press, 2016.

[26] R. Merletti and S. Muceli, "Tutorial. Surface EMG detection in space and time: Best practices," *J. Electromyogr. Kinesiol.*, vol. 49, Dec. 2019, Art. no. 102363.

[27] A. Del Vecchio, A. Holobar, D. Falla, F. Felici, R. M. Enoka, and D. Farina, "Tutorial: Analysis of motor unit discharge characteristics from high-density surface EMG signals," *J. Electromyogr. Kinesiol.*, vol. 53, Aug. 2020, Art. no. 102426.

[28] A. Ameri, M. A. Akhaee, E. Scheme, and K. Englehart, "Regression convolutional neural network for improved simultaneous EMG control," *J. Neural Eng.*, vol. 16, no. 3, Jun. 2019, Art. no. 036015.

[29] R. C. Sîmpetru, M. Osswald, D. I. Braun, D. S. Oliveira, A. L. Cakici, and A. Del Vecchio, "Accurate continuous prediction of 14 degrees of freedom of the hand from myoelectrical signals through convolutive deep learning," in *Proc. 44th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Jul. 2022, pp. 702–706.

[30] R. C. Sîmpetru et al., "Sensing the full dynamics of the human hand with a neural interface and deep learning," Dec. 2022, doi: 10.1101/2022.07.29.502064.

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

[32] A. L. Cakici et al., "A generalized framework for the study of spinal motor neurons controlling the human hand during dynamic movements," in *Proc. 44th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Jul. 2022, pp. 4115–4118.

[33] A. Mathis et al., "DeepLabCut: Markerless pose estimation of user-defined body parts with deep learning," *Nature Neurosci.*, vol. 21, no. 9, pp. 1281–1289, Sep. 2018.

[34] P. Karashchuk et al., "Anipose: A toolkit for robust markerless 3D pose estimation," *Cell Rep.*, vol. 36, no. 13, Sep. 2021, Art. no. 109730.

[35] A. Del Vecchio, A. Úbeda, M. Sartori, J. M. Azorín, F. Felici, and D. Farina, "Central nervous system modulates the neuromechanical delay in a broad range for the control of muscle force," *J. Appl. Physiol.*, vol. 125, no. 5, pp. 1404–1410, Nov. 2018.

[36] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. 33rd Conf. Neural Inf. Process. Syst.*, 2019, p. 12.

[37] W. Falcon and TPL Team, "PyTorch lightning," Mar. 2023, doi: 10.5281/zenodo.7688620.

[38] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[39] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," 2016, *arXiv:1607.08022*.

[40] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.

[41] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," 2016, *arXiv:1606.08415*.

[42] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, *arXiv:1711.05101*.

[43] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond," in *Proc. Int. Conf. Learn. Represent.*, Feb. 2018, pp. 1–7.

[44] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," 2017, *arXiv:1708.07120*.

[45] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. Wilson, "Averaging weights leads to wider optima and better generalization," in *Proc. 34th Conf. Uncertainty Artif. Intell.*, vol. 2, 2018, pp. 876–885.

[46] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2017.

[47] L. Campagnola et al., "VisPy: Interactive scientific visualization in Python," Tech. Rep., Nov. 2022.

[48] P. Ramachandran, "Mayavi: 3D visualization of scientific data in Python," Tech. Rep., May 2021.

[49] A. Del Vecchio et al., "Spinal motoneurons of the human newborn are highly synchronized during leg movements," *Sci. Adv.*, vol. 6, no. 47, Nov. 2020, Art. no. eabc3916.

[50] A. K. Clarke et al., "Deep learning for robust decomposition of high-density surface EMG signals," *IEEE Trans. Biomed. Eng.*, vol. 68, no. 2, pp. 526–534, Feb. 2021.