
Article:
Bao, T orcid.org/0000-0002-1103-2660, Zaidi, SAR orcid.org/0000-0003-1969-3727, Xie, S orcid.org/0000-0003-2641-2620 et al. (3 more authors) (2021) A deep Kalman filter network for hand kinematics estimation using sEMG. Pattern Recognition Letters, 143. pp. 88-94. ISSN 0167-8655

https://doi.org/10.1016/j.patrec.2021.01.001

Pattern Recognition Letters
journal homepage: www.elsevier.com

A Deep Kalman Filter Network for Hand Kinematics Estimation using sEMG

Tianzhe Bao[a,**], Syed Ali Raza Zaidi[a], Shengquan Xie[a], Pengfei Yang[b], Yihui Zhao[a], Zhiqiang Zhang[a]

[a] *School of Electrical and Electronic Engineering, University of Leeds, LS2 9JT, UK*
[b] *School of Computer Science and Technology, Xidian University, China*

ABSTRACT

In human-machine interfaces (HMI), deep learning (DL) techniques such as convolutional neural networks (CNN), long-short term memory networks (LSTM) and the hybrid CNN-LSTM framework have been exploited for hand kinematics estimation using surface electromyography (sEMG). However, these DL techniques only capture the relationship between sEMG and hand kinematics, but ignores the prior knowledge of the system. By contrast, Kalman filter (KF) can apply Kalman gain to combine the internal transition model and the observation model effectively. To this end, we propose a novel architecture named deep Kalman filter network (DKFN), in which we utilize CNN to extract high-level features from sEMG and employ a LSTM-based Kalman filter process (LSTM-KF) to conduct sequential regression. In particular, LSTM-KF adopts the computational graph of KF but estimates parameters of the transition/observation model and the Kalman gain from data using LSTM modules. With this process, the advantages of KF and LSTM can be exploited jointly. Experimental results demonstrate that the proposed DKFN can outperform CNN and CNN-LSTM in the sequential regression for wrist/fingers kinematics estimation.

1. Introduction

Surface electromyography (sEMG) signals are bioelectricity collected non-invasively by surface electrodes attached on the skin [1]. They are combined actions of motor unit action potentials (MUAPs) produced by muscle fibres along with neuromuscular activities, and can reflect the extent of muscle contractions ahead of actual motions [2]. Therefore it has been working as one of the main control sources in human-machine interfaces (HMI) such as intelligent prostheses or exoskeleton robotics [3]. To decode human intentions from sEMG more intuitively, regression-based approaches, including numerous linear/non-linear models and various sEMG features, have been widely investigated in the past decade [4, 5]. Different from other techniques such as the classifier-based pattern recognition [6], regression methods intend to estimate joint kinematics or kinetics in multiple degrees of freedom (DoFs) continuously [5, 7], and can enable the simultaneous and proportional control in HMI.

To further improve regression accuracies and robustness, two main deep learning (DL) techniques are now becoming prevalent in recent years: 1) convolutional neural networks (CNN) for feature extraction; 2) recurrent neural networks (RNN) for sequential regression. Due to the stochastic nature of sEMG and serious cross-talk among muscles in the upper limb, useful information can be easily buried in conventional hand-crafted features. To this end, CNN have been applied to extract features from raw sEMG automatically, where promising results can be found in both wrist kinetics and kinematics estimation [2, 3, 8]. One limitation of CNN is that it mainly focuses on spatial correlations among channels of sEMG but inherently ignore temporal dependencies between adjacent samples. In regression-based motion estimations, the targeted movements are continuous and sEMG segments should be regarded as time-series data. Since RNN, particularly the long-short term memory network (LSTM), shows great effectiveness in capturing long-term dependencies by learning contextual information from past model inputs, there is now a trend to implement RNN/LSTM in HMI [9, 10]. Inspired

e-mail: eltb@leeds.ac.uk (Tianzhe Bao)

by the advantages of CNN and RNN, hybrid structures have been preliminarily investigated to better exploit the CNN-based feature extraction and RNN-based sequence learning [11, 12]. Experiment results demonstrate that a CNN-LSTM hybrid framework achieved much higher accuracy than CNN and ML approaches [11]. However, conventional DL techniques, including CNN and RNN, can neither include prior knowledge of the system nor capture the uncertainty information. Therefore, they are easy to suffer from very noisy measurements which are quite common in neural activities [13].

Apart from data-driven techniques mentioned above, Kalman filters (KF) have also been employed to infer limb movements from neural recordings [14, 15]. KF uses the Kalman gain to determine the weights of the internal transition model and the observation model, thus it can track the system state throughout time using noisy measurements. However, KF could not handle the non-linear relationship between the neural activity and limb movements. Besides, it is hard to pre-determine the value of parameters in transition and measurement models. To overcome such limitations, many attempts have been made to directly learn models from training data with LSTM. For example, Coskun et al firstly proposed the LSTM-KF framework which integrated three LSTM modules into the KF to learn the transition, observation and noise models in pose estimation tasks [16]. Zhao et al proposed a Learning Kalman Network (LKN) to filter the car trajectory given a sequence of measurements. In this network, both the observation model and transition model were learned by feed-forward neural networks, whilst the Kalman Gain iteration was enhanced by a LSTM module [17]. More recently, Ju et al designed a Interaction-aware Kalman Neural Networks (IaKNN), in which they incorporated two LSTM modules for learning the time-varying process and measurement noises for the update step of the Kalman filter [18]. However, all these studies were conducted in computer-vision tasks, such as human pose reconstruction or trajectory prediction of self-driving cars. To our best knowledge, there has yet any research on sEMG-based motion estimation.

In this paper, we propose a novel architecture named Deep Kalman Filter Network (DKFN) to estimate hand motions using sEMG. As an expansion of CNN-LSTM framework [11], DKFN firstly utilizes CNN to extract high-level features from raw sEMG signals, and then employs LSTM-based Kalman filter process (LSTM-KF) to conduct sequential regression of CNN features. The main difference between DKFN and the conventional CNN-LSTM lies in LSTM-KF which integrates LSTM into KF for recursive learning, such that the advantages of KF and LSTM can be exploited jointly. In particular, LSTM-KF adopts the computational graph of KF but estimates parameters of the transition/observation model and the Kalman gain from data using LSTM modules. To validate the effectiveness of DKFN, eight healthy subjects were recruited to perform continuous wrist movements. Moreover, a public benchmark is further utilized to verify the generalization of DKFN in mapping sEMG to finger kinematics. Experimental results demonstrate that DKFN can outperform CNN and CNN-LSTM in the sequential regression for both wrist and finger kinematics estimation.

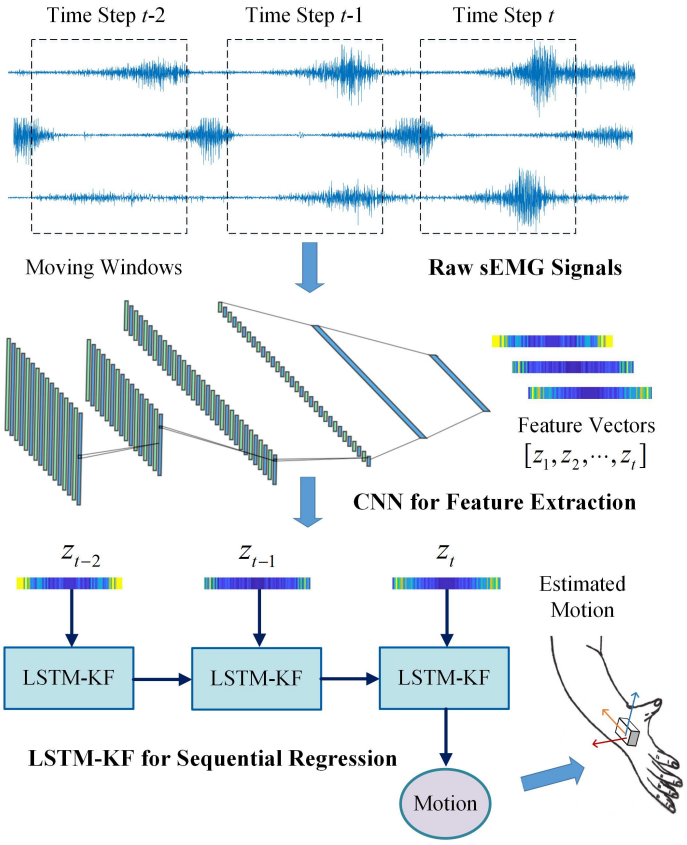

Fig. 1: Pipeline of DKFN in hand motion estimation using sEMG. It is noted that DKFN is composed two parts, i.e. a CNN for deep feature extraction and a LSTM-KF module for sequential regression.

The remainder of this paper is structured as follows. Section II describes the pipeline of the proposed DKFN. We firstly introduce the CNN-based feature extraction, the KF-based filtering and LSTM-based sequential regression, and then describe the design of LSTM-KF in detail. Section III introduces experimental setups of wrist/finger kinematics estimation and Section IV presents experimental results in these two tasks. In Section V the paper is concluded with some future research avenues discussed.

2. Methodology

2.1. Overview

The pipeline of DKFN for hand motion estimation is illustrated in Fig. 1. Sliding windows are firstly applied to split the multi-channel sEMG into successive sEMG matrices in different time steps. Then CNN is utilized to extract feature vectors $\mathbf{z}$ from sEMG matrices. Based on these features, a group of training sequences $[\mathbf{z}_{t-l+1}, \cdots, \mathbf{z}_t]$ (t denotes the time step and l is the length of this sequence) can be rebuilt for the sequential regression conducted by

LSTM-KF. In our design, LSTM-KF follows the computational graph of KF and estimates parameters of the transition/observation model from data using LSTM modules. Thus LSTM-KF is recursive over time and we can obtain a sequence of outcomes, i.e. $[\mathbf{x}_{t-l+1}, \cdots, \mathbf{x}_t]$ for $[\mathbf{z}_{t-l+1}, \cdots, \mathbf{z}_t]$. The final state $\mathbf{x}_t$ is taken as the regression output of this sample. As shown in Fig. 1, a many-to-one recursive structure was adopted in LSTM-KF. In order to estimate kinematics in every time step, the increment between training sequences was set to be one feature vector. In the following part, we will briefly introduce the CNN-based feature extraction and then elaborate the design of LSTM-KF.

2.2. Feature extraction

In this study we apply a single-stream CNN to extract features based on pre-processed sEMG. Specifically, a sliding window method is utilized to obtain sEMG matrix in the size of $1 \times L \times C$, where L denotes the window length and C represents the number of sensor channels. This format can enhance the exploitation of spatial-correlations among sEMG channels via convolutional operations. Since the spectrum of sEMG is observed to be more distinguishable than temporal signals between different motions, the fast Fourier transform (FFT) is applied on each channel of sEMG matrix.

Similar to [8], the presented CNN consists of 4 convolutional blocks (Conv Block) and 2 fully connected blocks (FC Block). Each Conv Block is composed of a convolutional layer, a batch normalization layer, a leaky ReLU layer, a max-pooling layer and a dropout layer. The convolutional layer uses a kernel size of 3, a boundary padding of 1 and the stride of 1. There are 16 kernels in the 1st and 2nd Conv Block, whilst 32 in the 3rd and 4th block. In each FC Block, the batch normalization layer, leaky ReLU layer and dropout layer are added subsequently to the fully connected layer. There are 100 hidden units in the 1st FC Block and 20 in the 2nd. Outputs of the 2nd FC Block will be utilized as features for sequential regression.

2.3. Design of LSTM-KF

Since the proposed LSTM-KF follows the computational graph of KF, we firstly introduce the typical workflow of KF in decoding upper limb kinematics from neural activities [14, 13]. Specially, the system dynamics is formulated as a transition model and an observation model:

$$\mathbf{x}_i = \mathbf{A}\mathbf{x}_{i-1} + \mathbf{w} \tag{1}$$

$$\mathbf{z}_i = \mathbf{H}\mathbf{x}_i + \mathbf{v} \tag{2}$$

where $\mathbf{z}_i$ is the CNN feature vector in the ith ($i \in [t-l+1, t]$) time step and $\mathbf{x}_i$ denotes the kinematic state of the hand. $\mathbf{A}$ is the state transition matrix, $\mathbf{H}$ is the measurement transformation matrix,$\mathbf{w}$ and $\mathbf{v}$ are the state and measurement uncertainties drawn from Gaussian distributions $N(\mathbf{0}, \mathbf{Q})$ and $N(\mathbf{0}, \mathbf{R})$.

With these parameters determined, KF utilizes an iterative feedback loop with a prediction step and an update step to fuse two models. In the prediction step, the mean and covariance of the current state are estimated by

$$\hat{\mathbf{x}}_i = \mathbf{A}\mathbf{x}_{i-1} \tag{3}$$

$$\hat{\mathbf{P}}_i = \mathbf{A}\mathbf{P}_{i-1}\mathbf{A} + \mathbf{Q} \tag{4}$$

where $\hat{\mathbf{x}}_i$ is the prior state estimate for the ith time step and $\hat{\mathbf{P}}_i$ denotes the prior state covariance matrix.

In the update step, the Kalman gain $\hat{\mathbf{K}}_i$ is calculated by

$$\hat{\mathbf{K}}_i = \hat{\mathbf{P}}_i\mathbf{H}^T\left(\mathbf{H}\hat{\mathbf{P}}_i\mathbf{H} + \mathbf{R}\right)^{-1} \tag{5}$$

Based on $\hat{\mathbf{K}}_i$ and the observed measurement $\mathbf{z}_i$, the mean and covariance of the posterior state estimate, i.e. $\mathbf{x}_i$ and $\mathbf{P}_i$, can be updated as

$$\mathbf{x}_i = \hat{\mathbf{x}}_i + \mathbf{K}_i\left(\mathbf{z}_i - \mathbf{H}\hat{\mathbf{x}}_i\right) \tag{6}$$

$$\mathbf{P}_i = \left(\mathbf{I} - \hat{\mathbf{K}}_i\mathbf{H}\right)\hat{\mathbf{P}}_i \tag{7}$$

where $\mathbf{I}$ is an identity matrix.

In conventional KF, parameters $\mathbf{A}$, $\mathbf{H}$, $\mathbf{Q}$ and $\mathbf{R}$ are constant matrices that need to be hand-designed under simplistic assumptions, which might be arbitrary and crude for many practical tasks [16]. Besides, the transition model and the observation model of KF could only capture linear relationships between sEMG signals and hand kinematics, whereas a high non-linearity has been verified in numerous literatures [2, 9, 10].

Similar to KF, LSTM is a network with loops inside. Different from KF, LSTM captures the non-linear dynamics between the input and the output with its weight matrices learned from data. The implementation of LSTM in sEMG-based kinematics estimation can be formulated as

$$\begin{aligned}
\boldsymbol{i}_i &= \delta\left(\boldsymbol{W}_i\left[\boldsymbol{s}_{i-1}, \mathbf{z}_i\right] + \boldsymbol{b}_i\right) \\
\boldsymbol{m}_i &= \delta\left(\boldsymbol{W}_m\left[\boldsymbol{s}_{i-1}, \mathbf{z}_i\right] + \boldsymbol{b}_m\right) \\
\boldsymbol{o}_i &= \delta\left(\boldsymbol{W}_o\left[\boldsymbol{s}_{i-1}, \mathbf{z}_i\right] + \boldsymbol{b}_o\right) \\
\boldsymbol{c}_{\boldsymbol{i}} &= \boldsymbol{i}_{\boldsymbol{i}} \odot \tanh\left(\boldsymbol{W}_c\left[\boldsymbol{s}_{i-1}, \mathbf{z}_i\right] + \boldsymbol{b}_c\right) + \boldsymbol{m}_i \odot \boldsymbol{c}_{i-1} \\
\boldsymbol{s}_i &= \boldsymbol{o}_i \odot \tanh\left(\boldsymbol{c}_i\right) \\
y_i &= \boldsymbol{W}_y\boldsymbol{s}_i + \boldsymbol{b}_y
\end{aligned} \tag{8}$$

where $\boldsymbol{s}_i$ is the hidden state at time-step i, $\boldsymbol{c}_i$ is the activation vector, $\boldsymbol{i}_i$ is the input gate, $\boldsymbol{m}_i$ is the forget gate, $\boldsymbol{o}_i$ is the output gate, δ is the logistic sigmoid function, $\boldsymbol{W}$ is the weight matrix in each gate and layer, $\boldsymbol{b}$ is the corresponding bias vector and $\odot$ is the scalar product. The initial state $(\boldsymbol{h}_0, \boldsymbol{c}_0)$ will be settled after model training for subsequent predictions.

However, LSTM mainly exploits the relationship between hand kinematics and sEMG features, but can neither include prior knowledge of the system nor capture the uncertainty information. In this study, the proposed LSTM-KF intends to solve these limitations by combining merits of KF and LSTM. Specifically, LSTM-KF follows

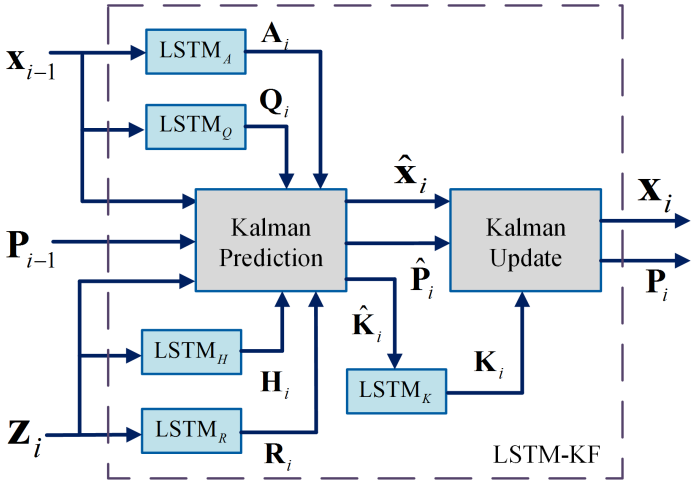

Fig. 2: The feed-forward architecture of LSTM-KF in one loop.

the computational graph of KF described in Eq. (3)-(7), but $\mathbf{A}$, $\mathbf{H}$, $\mathbf{Q}$ and $\mathbf{R}$ are designed to be learnable from data on the basis of several LSTM modules [16].

In the transition model, $\mathbf{A}_i$ and $\mathbf{Q}_i$ are produced by LSTM modules using previous state estimate $\mathbf{x}_{i-1}$:

$$\mathbf{A}_i, \mathbf{s}_i^A = LSTM_A\left(\mathbf{x}_{i-1}, \mathbf{s}_{i-1}^A\right) \tag{9}$$

$$\mathbf{Q}_i, \mathbf{s}_i^Q = LSTM_Q\left(\mathbf{x}_{i-1}, \mathbf{s}_{i-1}^Q\right) \tag{10}$$

where $LSTM_A$ and $LSTM_Q$ are LSTM modules for the approximation of $\mathbf{A}_i$ and $\mathbf{Q}_i$, whilst $\mathbf{s}_{i-1}^A$ and $\mathbf{s}_{i-1}^Q$ denote the hidden states of $LSTM_A$ and $LSTM_Q$, respectively.

In the observation model, we use the CNN feature $\mathbf{z}_i$ as measurement to produce $\mathbf{H}_i$ and $\mathbf{R}_i$:

$$\mathbf{H}_i, \mathbf{s}_i^H = LSTM_H\left(\mathbf{z}_i, \mathbf{s}_{i-1}^H\right) \tag{11}$$

$$\mathbf{R}_i, \mathbf{s}_i^R = LSTM_R\left(\mathbf{z}_i, \mathbf{s}_{i-1}^R\right) \tag{12}$$

The obtained $\mathbf{A}_i$, $\mathbf{R}_i$, $\mathbf{Q}_i$ and $\mathbf{R}_i$ are then fed into the prediction step and then the update step of KF to compute $\mathbf{x}_{i-1}$ and $\mathbf{P}_{i-1}$. The feed-forward architecture of LSTM-KF in one loop is depicted in Fig. 2. As we can see, in LSTM-KF, both the transition/observation matrices and the corresponding covariance matrices can change through time dynamically. Moreover, as suggested by [17], $\hat{\mathbf{K}}_i$ in Eq. (5) is defined as the intermediate gain, and an extra LSTM module is exploited to learn the final gain $\mathbf{K}_i$ based on $\hat{\mathbf{K}}_i$:

$$\mathbf{K}_i, \mathbf{s}_i^K = LSTM_K\left(\hat{\mathbf{K}}_i, \mathbf{s}_{i-1}^K\right) \tag{13}$$

2.4. Model training

Following the training approaches in [16], CNN and LSTM-KF are also trained separately in this study. To train CNN, a regression layer is attached to the presented CNN architecture to complete a supervised learning process, in which inputs are sEMG matrices and labels are measured kinematics. After this step, feature vectors can be extracted from the second FC layer of CNN to construct training sequences LSTM-KF. From Eq. (3)-(13) we can see that feed-forward computation of LSTM-KF is conducted with iterative loops, and that all learnable weights come from LSTM modules. The standard Euclidean $L2$ loss is employed in the loss function of LSTM-KF, with a regularization term to mitigate against over-fitting:

$$L(\boldsymbol{W}) = \sum_{j=1}^{N} \sum_{i=t-l+1}^{t} \left\| \mathbf{x}_p^{ij} - \mathbf{x}_{gt}^{ij} \right\|_2 + \lambda \|\boldsymbol{W}\|_2 \tag{14}$$

where N is the number of training sequences in a training-batch, $\mathbf{x}_p^{ij}$ and $\mathbf{x}_{gt}^{ij}$ are predicted and ground-truth kinematics. λ is the scaling factor for regularization.

Fig. 3: The placement of electrodes and markers in wrist kinematics estimation. The current gesture was regarded as the neutral position in the continuous wrist movement.

3. Experimental methods

In this study, we mainly focused on the joint kinematics estimation of wrist and fingers. Two data sources were employed to validate performances of LSTM-KF. The first data source was obtained from our experiments, in which wrist kinematics (flexion/extension) of eight healthy subjects were captured and reconstructed. By contrast, the second source is a public benchmark which involves multi-DoF finger motions (index flexion, middle flexion, ring flexion) from ten able-bodied subjects and two amputees.

3.1. Wrist kinematics experiment

Approved by the MaPS and Engineering joint Faculty Research Ethics Committee of University of Leeds, UK (MEEC 18-002), six males and two females (aged 25 to 31) participated in this experiment. A written informed consent was obtained from each subject. As shown in Fig. 3, subjects are asked to seat on the armchair, with torso fully straight and forearm relaxed. The current position of hand was set as the neutral position. During data collection, participants were performed wrist flexion/extension following a continuous cycle trial: the wrist was rotated from neutral position to the flexion direction, it was then moved back to the extension direction and finally returned to neutral position. Five repetitive trials were conducted

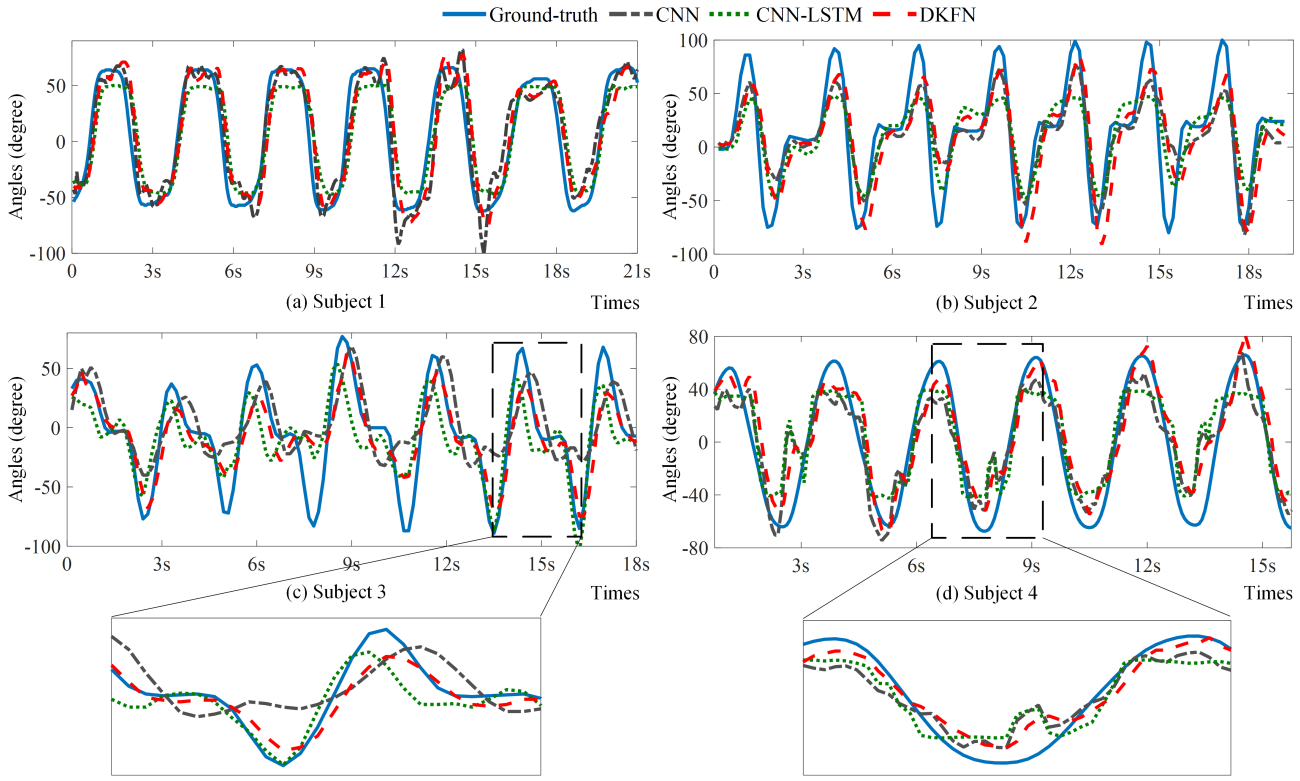

Fig. 4: Performances of CNN, CNN-LSTM and DKFN in wrist kinematics estimation.

by each participate and each trial lasted for about 20s. A three minutes break was given between each trial to prevent muscle fatigue.

In our experiment, we used Delsys TrignoTM system to record sEMG signals. Avanti electrodes are placed over five wrist muscles (Flexor Carpi Radialis, lexor Carpi Ulnaris, Extensor Carpi Radialis Longus, Extensor Carpi Radialis Brevis and and Extensor Carpi Ulnaris) over right forearm. The sampling rate of was set as 2000 Hz. To capture wrsit movements through the motion capture system (Vicon Motion Systems Ltd. UK), 16 reflective markers were placed on the subject's right upper limb. As illustrated in Fig. 3, markers were allocated over the spinous process of the 7th and the 10th thoracic vertabra, right scapula, xiphoid, acromio-clavicular joint, clavicle, lateral/medial humerus medial epicondyle, right radial/ulnar styloid, middle forearm and the right third metacarpus. The sampling rate of Vicon Motion System was 250 Hz and the synchronization of the kinematic data and sEMG were conducted using a trigger module. The Vicon upper limb model were applied to calculate wrist joint angles as the measured angles or ground truth.

3.2. Finger kinematics experiment

In order to verify the generalization of LSTM-KF, the finger kinematics estimation was conducted using data from the 8th Ninapro database [19]. Ten able-bodied subjects (nine male, one female) and two male, right-hand transradial amputees were recruited. The Delsys TrignoTM system was utilized to collect sEMG data from participants' right forearm at a rate of 1111 Hz. The finger kinematic data were recorded with a dataglove (Cyberglove 2, 18-DOF model) worn on the left hand. During data acquisition, participants were asked to perform bilateral mirrored movements, in which nine hand motions were activated. Each motion execution lasted around 7s with a 3s interval. Three trials were recorded for each participant. The first two trails comprised ten repetitions of each hand movement and the third trial comprised two repetitions. A ten minutes break was given to participants after each trial.

3.3. Data pre-processing

Raw sEMG signals obtained from the front-end acquisition equipment are generally noisy because of electromagnetic radiation, motion artefact and instability of signals, etc. [20]. In particular, noise produced by motion artefact is in the range of 0-20 Hz. Thus signals below 20 Hz are unstable and cannot provide a reliable contribution to sEMG [21]. A simple method is to filter it out with a high-pass filter. In addition, since energy of sEMG is mainly blow 450 Hz, a low-pass filter is often used to remove unwanted noise with high frequencies [22]. As suggested by these studies, we applied a 3rd order Butterworth band-pass filter (20-450 Hz) to reduce noise of raw sEMG.

To complete feature extraction, the size of sliding windows was set to be 100ms length with 20ms increment (80% overlap). Thus the size of sEMG matrix ($1 \times L \times C$) was $1 \times 199 \times 5$. Since subjects were asked to rotate the wrist in a comparatively low speed, the label of a sample

was obtained by computing the mean value of angles (captured by Vicon) within the processing window. To implement sequential regression, the time duration of a training sequence was set as 300ms, resulting in 10 time steps in $[\mathbf{z}_{t-l+1}, \cdots, \mathbf{z}_t]$, i.e. $l = 10$. As for the labels in finger kinematics estimation, glove measurements were also averaged within a processing window and then converted into finger kinematics via a linear mapping [19]. The sliding windows were 200ms in length with 50ms increment, and the time duration of a training sequence was set as 500ms.

3.4. Hyper-parameter settings

The training of CNN and LSTM modules is implemented using Tensorflow backend. Specifically, CNN was trained in a 32 sized mini-batch for 100 epochs via adaptive moment estimation (ADAM). The dynamic learning rate was 0.0001 with a decay rate of 0.001 for each iteration. The slope scale of leaky ReLU layers was set as 0.1. The max-pooling layer used a pool size of 3, whilst the dropout rate is set to be 30%. As for LSTM modules, weights were initialized using Xavier initialization and then trained in a 32 sized mini-batch for 100 epochs via ADAM. The initial learning rate was set as 0.001 and retain 0.99 in each epoch. The regularization factor λ was 0.01. In general, the number of hidden layers in LSTM will affect the regression performance. In our experiment, we empirically observed that 20 hidden layers could be a good trade-off for two datasets since more layers would result in a much heavier computational load with limited improvement in model accuracy.

4. Results and discussion

In this study, performances of DKFN were evaluated using inter-session results. In the wrist kinematics experiment, the model was trained in the first four trials and tested in the last trial. To validate the finger kinematics estimation, we used the first two sessions for model training the last session for testing. Herein, coefficient of determination (R^2) and root mean square error (RMSE) served as the performance metric. In specific, R^2 indicates how the estimated curve is related to measured joint motion, whilst RMSE computes the differences in amplitude between estimated kinematics and measurement. Herein we chose CNN and CNN-LSTM as baseline methods for model comparison.

4.1. Wrist kinematics estimation

Fig. 4 illustrates the estimation performances of CNN, CNN-LSTM and DKFN in Subject 1-4. As we can see, wrist motions can be effectively reconstructed from sEMG signals using three methods. In each subject, the trajectory reconstructed by DKFN (red dashed line) is closer to the ground truth (blue solid line) than other two trajectories. Another interesting observation in Fig. 4 is that trajectories of both CNN-LSTM and DKFN are comparatively smoother than the trajectory of CNN. A main reason is that the sequential regression conducted by LSTM and LSTM-KF can reduce some fluctuations by exploiting temporal dependencies among adjacent sEMG features. Different from conventional smoothing techniques such as the moving average, LSTM and LSTM-KF do not cause lags. In terms of the mappings of sEMG with wrist kinematics, LSTM and LSTM-KF are capable of capturing some of those buried in CNN features. As shown in Fig. 4 (c), CNN-LSTM and DKFN managed to track target motions performed by Subject 3 between 13s-16s, whereas CNN only captured half of that trajectory.

Table 1: R^2 and RMSE of CNN, CNN-LSTM and DKFN in wrist kinematics estimation. S1-S8 denote Subject 1-8 respectively.

	R^2			RMSE		
	CNN	CNN-LSTM	DKFN	CNN	CNN-LSTM	DKFN
S1	0.87	0.89	0.92	20.13	19.49	17.53
S2	0.66	0.77	0.83	28.25	23.41	20.20
S3	0.47	0.57	0.69	26.24	23.20	16.20
S4	0.61	0.74	0.86	28.62	21.57	15.57
S5	0.84	0.87	0.90	14.14	12.79	11.63
S6	0.71	0.95	0.93	23.16	9.28	11.25
S7	0.82	0.86	0.90	18.93	16.85	14.22
S8	0.75	0.89	0.94	18.97	12.49	9.36

To compare estimation performances quantitatively, Table 1 summarizes the R^2 and RMSE of three methods in all subjects. The average performances (mean±standard deviation) of CNN, CNN-LSTM and DKFN are 0.72±0.13, 0.82±0.12, 0.87±0.08 for R^2 and 22.31±5.14, 17.39±5.39, 14.50±3.61 for RMSE. As we can see, DKFN achieves much larger R^2 and smaller RMSE than other two methods. The outperformance of DKFN can be more evident in Subject 3-4. A special case in our experiment is Subject 6, where CNN-LSTM outperforms DKFN slightly. In fact R^2 values of 0.95 (CNN-LSTM) and 0.93 (DKFN) are actually quite close, and the predicted curves can both catch the ground-truth with very limited errors. It can be inferred that in this subject the extracted CNN features can be very well matched with the sequential regression conducted by vanilla LSTM or LSTM-KF.

4.2. Finger kinematics estimation

Different from the wrist kinematics estimation above, finger motions were activated in multi-DoFs simultaneously. In this study, we empirically trained/tested all three models using kinematics data in each DoF (index/middle/ring flexion) separately. Fig. 5 demonstrates the performances of CNN, CNN-LSTM and DKFN for estimating finger positions in an able-bodied subject (AB9). As we can see, three methods are capable of capturing measured trajectories in most cases. Similar to wrist kinematics estimations, herein LSTM and LSTM-KF can help to generate smoother trajectories and mitigate some abrupt fluctuations or deteriorations of CNN, such as the last peak of curves in the middle flexion of AB9.

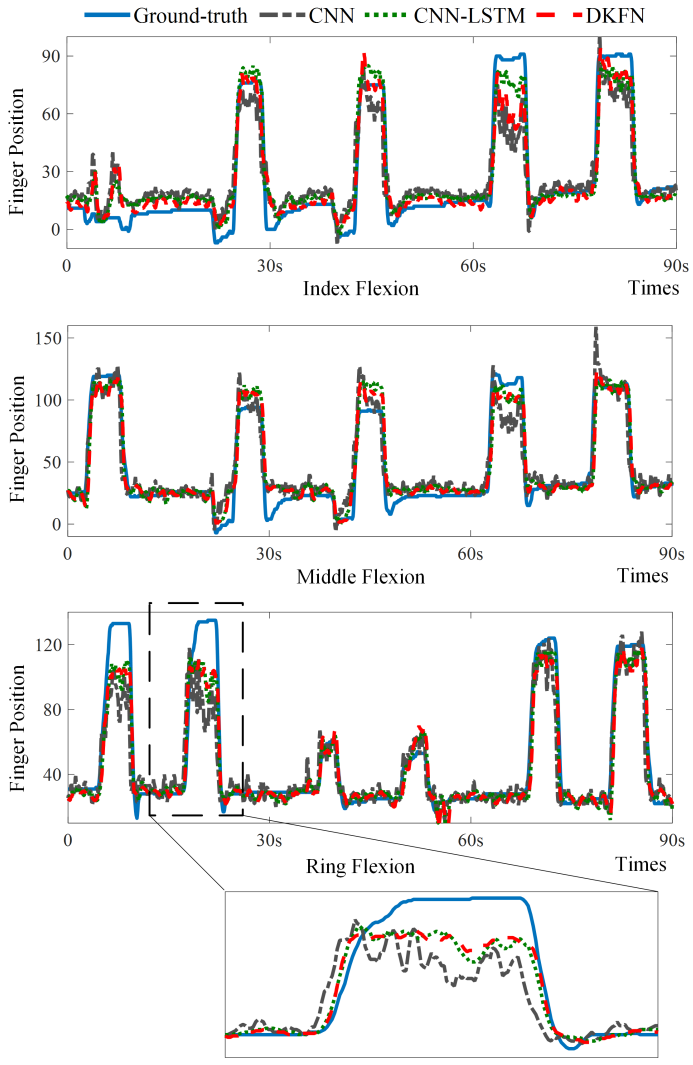

Fig. 5: Performances of CNN, CNN-LSTM and DKFN in finger kinematics estimation.

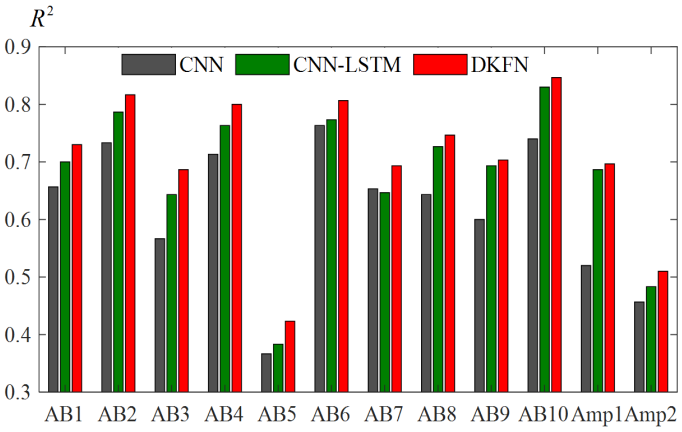

Fig. 6: Average R^2 of CNN, CNN-LSTM and DKFN among three DoFs in each subject.

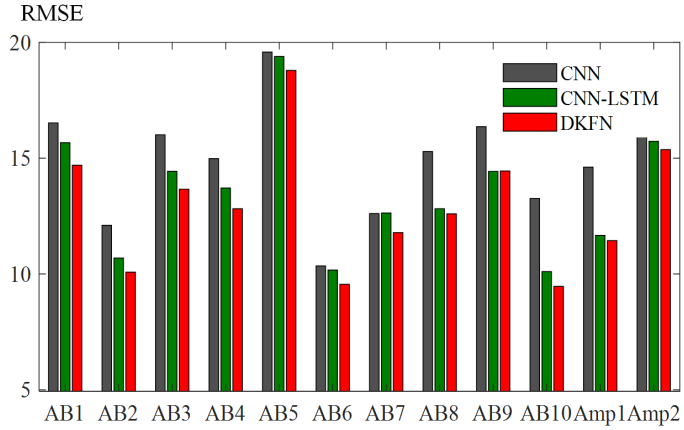

Fig. 7: Average RMSE of CNN, CNN-LSTM and DKFN among three DoFs in each subject.

Fig. 6 and Fig. 7 demonstrate the average model performances among all DoFs in each subject. The average performances of CNN, CNN-LSTM and DKFN among all DoFs in able-bodied subjects are 0.64±0.14, 0.69±0.15, 0.73±0.14 for R^2 and 14.71±3.40, 13.40±3.60, 12.78±3.55 for RMSE. For finger kinematics of amputees, statistical results are 0.57±0.15, 0.66±0.17, 0.68±0.16 for R^2 and 14.65±2.31, 12.50±3.20, 12.10±3.18 for RMSE. From these figures we can observe two interesting results: 1) compared with CNN, the sequential regression conducted by CNN-LSTM or DKFN improves estimation accuracies significantly; 2) through embedding LSTM modules in the KF computational graph, LSTM-KF further enhances the accuracy of sequential regression. Besides, due to the complexity of finger motions, average performances of three models are inferior to those in wrist kinematics estimation. However, these models can still achieve high accuracies in many subjects, such as AB2, AB4 and AB6 (in ring flexion of AB2, R^2 of CNN, CNN-LSTM and DKFN can reach 0.82, 0.90 and 0.91).

4.3. Effects of sequence length

The length of training/testing sequences determines the number of deep feature vectors involved in the sequential regression. In general, a too short sequence cannot provide sufficient information for model learning, whereas a too long sequence will significantly reduce the computational efficiency. Thus, a trade-off is of importance to enhance the practical utilization of DKFN in myoelectric control. Herein, we validated the influence of sequence length in wrist/finger kinematics estimation. Five different time-steps were selected for comparison, i.e. 1, 5, 10, 20, 50. These time-steps correspond to the sequences lasting 120ms, 200ms, 300ms, 500ms, 1100ms in the wrist dataset and 250ms, 450ms, 700ms, 1200ms 2700ms in the finger dataset. Fig. 8 illustrated the average estimation results of DKFN among all subjects using varied length of sequences. As we can see, in both datasets the model performances are overall improved with the increase of sequence length, but improvements will be limited when the length becomes comparatively larger. Therefore, it can be inferred that a sequence lasting 300-500ms may work properly in wrist kinematics estimation, whilst a length of 500-800ms might be suitable for the finger dataset.

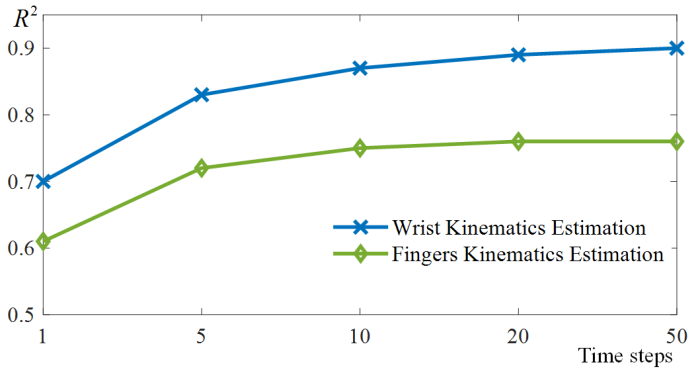

Fig. 8: Average estimation results of DKFN among all subjects with varied length of sequences.

5. Conclusion and future work

This paper proposed DKFN which utilizes a novel structure, i.e. LSTM-KF, to conduct sequential regression for hand kinematics estimation in HMI. With LSTM modules integrated into the computational graph of KF, parameters of the transition/observation model and Kalman gain can be learned from data effectively. Since LSTM-KF combines the non-linear transform property of LSTM with the probabilistic fusion mechanism of KF, it outperforms conventional LSTM in the sequential regression of CNN features extracted from sEMG in wrist/finger kinematics estimation. In the future work, we will further evaluate our model using data from the elbow and shoulder. The fusion of sEMG and inertia measurement unit will also be investigated to improve model robustness in more complicated scenarios.

Acknowledgments

We would like to thank all participants who were recruited in experiments for wrist kinematics estimation.

References

[1] MA Cavalcanti Garcia and TMM Vieira. Surface electromyography: Why, when and how to use it. *Revista andaluza de medicina del deporte*, 4(1):17–28, 2011.

[2] Wei Yang, Dapeng Yang, Yu Liu, and Hong Liu. Decoding simultaneous multi-dof wrist movements from raw emg signals using a convolutional neural network. *IEEE Transactions on Human-Machine Systems*, 49(5):411–420, 2019.

[3] Wentao Wei, Yongkang Wong, Yu Du, Yu Hu, Mohan Kankanhalli, and Weidong Geng. A multi-stream convolutional neural network for semg-based gesture recognition in muscle-computer interface. *Pattern Recognition Letters*, 2017.

[4] Koushik Bakshi, M Manjunatha, and CS Kumar. Estimation of continuous and constraint-free 3 dof wrist movements from surface electromyogram signal using kernel recursive least square tracker. *Biomedical Signal Processing and Control*, 46:104–115, 2018.

[5] Jie Liu, Yupeng Ren, Dali Xu, Sang Hoon Kang, and Li-Qun Zhang. Emg-based real-time linear-nonlinear cascade regression decoding of shoulder, elbow and wrist movements in able-bodied persons and stroke survivors. *IEEE Transactions on Biomedical Engineering*, 2019.

[6] Jinxian Qi, Guozhang Jiang, Gongfa Li, Ying Sun, and Bo Tao. Surface emg hand gesture recognition system based on pca and grnn. *Neural Computing and Applications*, pages 1–9, 2019.

[7] Ahmed W Shehata, Erik J Scheme, and Jonathon W Sensinger. Evaluating internal model strength and performance of myoelectric prosthesis control strategies. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 26(5):1046–1055, 2018.

[8] Manfredo Atzori, Matteo Cognolato, and Henning Müller. Deep learning with convolutional neural networks applied to electromyography data: A resource for the classification of movements for prosthetic hands. *Frontiers in neurorobotics*, 10:9, 2016.

[9] Fernando Quivira, Toshiaki Koike-Akino, Ye Wang, and Deniz Erdogmus. Translating semg signals to continuous hand poses using recurrent neural networks. In *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, pages 166–169. IEEE, 2018.

[10] Ali Samadani. Gated recurrent neural networks for emg-based hand gesture classification. a comparative study. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1–4. IEEE, 2018.

[11] Tianzhe Bao, Syed Ali Raza Zaidi, Shengquan Xie, Pengfei Yang, and Zhiqiang Zhang. A cnn-lstm hybrid model for wrist kinematics estimation using surface electromyography. *IEEE Transactions on Instrumentation and Measurement*, 2020.

[12] Miguel Simão, Pedro Neto, and Olivier Gibaru. Emg-based online classification of gestures with recurrent neural networks. *Pattern Recognition Letters*, 128:45–51, 2019.

[13] Wing-kin Tam, Tong Wu, Qi Zhao, Edward Keefer, and Zhi Yang. Human motor decoding from neural signals: a review. *BMC Biomedical Engineering*, 1(1):22, 2019.

[14] David J Warren, Spencer Kellis, Jacob G Nieveen, Suzanne M Wendelken, Henrique Dantas, Tyler S Davis, Douglas T Hutchinson, Richard A Normann, Gregory A Clark, and V John Mathews. Recording and decoding for neural prostheses. *Proceedings of the IEEE*, 104(2):374–391, 2016.

[15] Elizaveta Okorokova, Mikhail Lebedev, Michael Linderman, and Alex Ossadtchi. A dynamical model improves reconstruction of handwriting from multichannel electromyographic recordings. *Frontiers in neuroscience*, 9:389, 2015.

[16] Huseyin Coskun, Felix Achilles, Robert DiPietro, Nassir Navab, and Federico Tombari. Long short-term memory kalman filters: Recurrent neural estimators for pose regularization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5524–5532, 2017.

[17] Cheng Zhao, Li Sun, Zhi Yan, Gerhard Neumann, Tom Duckett, and Rustam Stolkin. Learning kalman network: A deep monocular visual odometry for on-road driving. *Robotics and Autonomous Systems*, 121:103234, 2019.

[18] Ce Ju, Zheng Wang, Cheng Long, Xiaoyu Zhang, Gao Cong, and Dong Eui Chang. Interaction-aware kalman neural networks for trajectory prediction. *arXiv preprint arXiv:1902.10928*, 2019.

[19] Agamemnon Krasoulis, Sethu Vijayakumar, and Kianoush Nazarpour. Effect of user adaptation on prosthetic finger control with an intuitive myoelectric decoder. *Frontiers in neuroscience*, 13:891, 2019.

[20] Mamun Bin Ibne Reaz, M Sazzad Hussain, and Faisal Mohd-Yasin. Techniques of emg signal analysis: detection, processing, classification and applications. *Biological procedures online*, 8(1):11–35, 2006.

[21] Carlo J De Luca, L Donald Gilmore, Mikhail Kuznetsov, and Serge H Roy. Filtering the surface emg signal: Movement artifact and baseline noise contamination. *Journal of biomechanics*, 43(8):1573–1579, 2010.

[22] Silvestro Micera, Jacopo Carpaneto, and Stanisa Raspopovic. Control of hand prostheses using peripheral information. *IEEE reviews in biomedical engineering*, 3:48–68, 2010.