NeuroMotion: Open-source Simulator with Neuromechanical and Deep Network Models to Generate Surface EMG signals during Voluntary Movement

Shihan Ma[1,2], Irene Mendez Guerra[1], Arnault Hubert Caillet[1], Jiamin Zhao[2], Alexander Kenneth Clarke[1], Kostiantyn Maksymenko[3], Samuel Deslauriers-Gauthier[3,4], Xinjun Sheng[2,5], Xiangyang Zhu[2,5], Dario Farina[1*]

1 Department of Bioengineering, Imperial College London, London, UK
2 State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai, China
3 Neurodec, Sophia Antipolis, France
4 Inria Centre at Université Côte d'Azur, Nice, France
5 Meta Robotics Institute, Shanghai Jiao Tong University, Shanghai, China

* d.farina@imperial.ac.uk

Abstract

Neuromechanical studies investigate how the nervous system interacts with the musculoskeletal (MSK) system to generate volitional movements. Such studies have been supported by simulation models that provide insights into variables that cannot be measured experimentally and allow a large number of conditions to be tested before the experimental analysis. However, current simulation models of electromyography (EMG), a core physiological signal in neuromechanical analyses, are mainly limited to static contractions and cannot fully represent the dynamic modulation of EMG signals during volitional movements. Here, we overcome these limitations by presenting NeuroMotion, an open-source simulator that provides a full-spectrum synthesis of EMG signals during voluntary movements. NeuroMotion is comprised of three modules. The first module is an upper-limb MSK model with OpenSim API to estimate the muscle fibre lengths and muscle activations during movements. The second module is BioMime, a deep neural network-based EMG generator that receives nonstationary physiological parameter inputs, such as muscle fibre lengths, and efficiently outputs motor unit action potentials (MUAPs). The third module is a motor unit pool model that transforms the muscle activations into discharge timings of motor units. The discharge timings are convolved with the output of BioMime to simulate EMG signals during the movement. Here we also provide representative applications of NeuroMotion. We first show how simulated MUAP waveforms change during different levels of physiological parameter variations and different movements. We then show that the synthetic EMG signals during two-degree-of-freedom hand and wrist movements can be used to augment experimental data for regression. Ridge regressors trained on the synthetic dataset were directly used to predict joint angles from experimental data. NeuroMotion is the first full-spectrum EMG generative model to simulate human forearm electrophysiology during voluntary hand, wrist, and forearm movements. All intermediate variables are available, which allows the user to study cause-effect relationships in the complex neuromechanical system, fast iterate algorithms before collecting experimental data, and validate algorithms that estimate non-measurable parameters in experiments. We expect this full-spectrum model will complement experimental approaches and facilitate neuromechanical research.

Author summary

Neuromechanical studies investigate how the nervous system and musculoskeletal system interact to generate movements. Such studies heavily rely on simulation models, which provide non-measurable variables to complement the experimental analyses. However, the simulation models of surface electromyography (EMG), the core physiological signal widely used in neuromechanical analyses, are limited to static conditions. We bridged this gap by proposing NeuroMotion, the first full-spectrum EMG simulator that can be used to generate EMG signals during voluntary movements. NeuroMotion integrates a musculoskeletal model, a neural network-based EMG generator, and an advanced motoneuron model. With representative applications of this simulator, we show that it can be used to investigate the variabilities of EMG signals during voluntary movement. We also demonstrate that the synthetic signals generated by NeuroMotion can be used to augment experimental data for regressing joint angles. We expect the functionality provided by NeuroMotion, which is provided open-source, will stimulate progress in neuromechanics.

Introduction

Human neuromechanics is the discipline that combines neuroscience and biomechanics to reach a fundamental understanding of the interactions between the nervous, muscular, and skeletal systems during human movements [1]. It allows us to uncover the functions and mechanisms of the nervous system under the production of movements [2, 3], by studying the movements from the perspective of their neural control [4]. Neuromechanical investigations are also important for developing decoding methods by providing the link between neural activities and behaviours. The decoding methods can be used to identify human intent from neural signals generated during a specific behaviour (neural interfaces). Neuromechanical studies have therefore allowed us to address problems ranging from motor control [4], rehabilitation engineering [5], and human-machine interfaces [6].

Due to the complex interplay of the nervous, muscular, and skeletal systems, researchers have used multiple simulation tools to explore certain aspects of human neuromechanics. A series of models that describe the neuron structures and functions have been proposed, including the classical Hodgkin-Huxley model [7]. At a macro scale, several analytical and numerical electromyography (EMG) models have been used to study the electrical outputs produced by skeletal muscles upon activations of populations of neurons [8–11]. In parallel, multiple simulation platforms, such as OpenSim [12, 13], have been widely used to study human body movements during dynamic simulations in biomechanical studies. However, the electrical outputs are rarely simulated together with the biomechanical system during voluntary movements.

OpenSim allows the users to include EMG signals as an additional input for better estimating the musculo-tendon dynamics and parameters but does not forwardly generate EMG signals during a movement [12]. Fuglevand's model has been widely used to study muscle force, motoneuron activities, and EMG signals mainly under isometric contractions [14]. A recurrent neural network has been recently used as a black box to convert motions to the downsampled and smoothed muscle activities without knowing the internal parameters of the system [15]. One primary reason for the lack of an integrated and precise EMG simulator feasible for voluntary movements is the absence of models that link EMG generation to movement biomechanics. Another reason is that current advanced EMG models are not efficient enough to adapt to the non-stationary physiological parameters during a voluntary movement. These two challenges have been addressed by our recently released EMG model, BioMime [16], which is a conditional generative model that takes the physiological parameters as inputs and outputs the dynamic motor unit action potential (MUAP) signals efficiently.

Using BioMime as a key module, here we propose NeuroMotion, an open-source EMG generative model that replicates the full-spectrum generation of electrophysiological signals during dynamic contractions. EMG signals are produced by the mixed convolutions of the MUAPs (electrical potential templates generated when one motoneuron is activated) and spike trains (continuous discharge timings of a motoneuron) of all active motor units during a movement. Correspondingly, NeuroMotion consists of three modules. The first module is an upper-limb musculoskeletal (MSK) model with OpenSim API for defining and visualising the movement and estimating the muscle fibre lengths and muscle activations during the movement. The muscle fibre lengths are then utilised by the second module, BioMime, to simulate the dynamic MUAPs during the movement. The third module is a motor unit pool model that receives the neural inputs derived from the muscle activations and outputs stimulations to each muscle in the format of spike trains. An overview of NeuroMotion is shown in Figure 1. With NeuroMotion, users can synthesise large EMG datasets under a vast repertoire of hand and wrist movements with all intermediate variables available. One potential application is to fast iterate regression and classification algorithms on a synthetic dataset before collecting experimental data (this data augmentation example will be used later in the Results). Another application is the validation of information extraction algorithms (e.g., EMG decomposition algorithms) when experimental data lacks the ground truth. Data, codes, and instructions are available at https://github.com/shihan-ma/NeuroMotion.

Methods

Overview

NeuroMotion is a surface EMG generative model that is designed to simulate EMG signals during voluntary human upper limb movements. NeuroMotion takes the kinematics of human hand, wrist, and forearm as inputs and outputs the synthetic surface EMG signals that correspond to the myoelectric activities responsible for the movement. Three modules are incorporated to complete this full-spectrum simulation, including an upper-limb MSK model with OpenSim API, BioMime, and a Motor Unit Pool model. Intermediate variables (changes in physiological parameters, MUAPs, and spike trains) are available in this hierarchical and modular simulation. Details of the three key modules are described in Section Core Modules. An overview of the workflow of NeuroMotion is shown in Figure 1.

Core Modules

MSK Model with OpenSim

OpenSim is an open-source software platform that is used for modelling and analysing neuromusculoskeletal systems [12, 13]. With this platform, researchers can build MSK models, simulate dynamic movements, and perform motion analysis for biomechanical studies that advance movement science. Here, we use OpenSim as a bridge between movements and physiological system states for three purposes: (1) to define and visualise movements of MSK models, (2) to track the changes in muscle fibre lengths during the movement, and (3) to predict the muscle activations across muscles.

Since we mainly focus on simulating EMG signals from the human forearm during hand, wrist, and forearm movements, we chose the ARMs Wrist and Hand Model [17] (ARMs in abbreviation) from the large database of opens-source OpenSim models as the

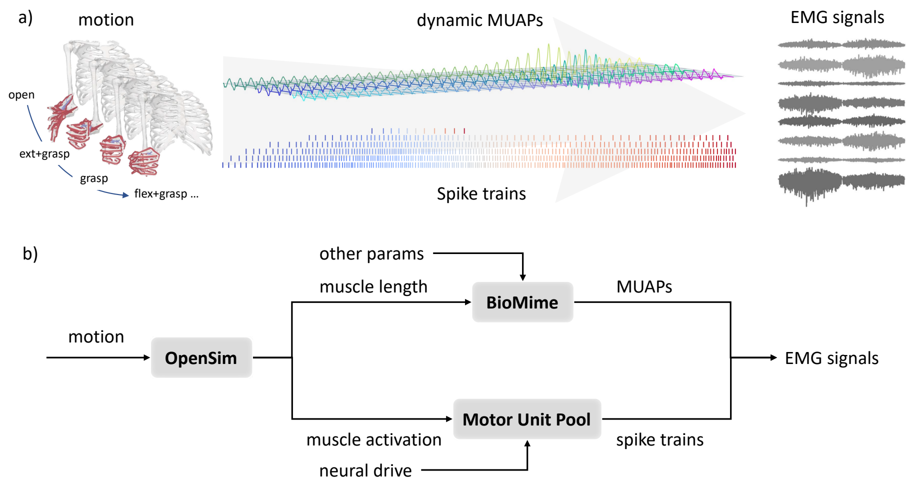

Fig 1. Overview of NeuroMotion. a), NeuroMotion provides an integrated generative model to simulate forearm surface EMG signals during voluntary hand and wrist movements. b), The movements are defined by the kinematics of an upper limb MSK model. Muscle fibre lengths during the movement are estimated by **OpenSim** and imported into **BioMime** to simulate dynamic MUAPs, which are the continuously changing electrical potential templates generated by the activations of the motor units. Neural inputs to each muscle, which can be derived from the normalised muscle activations from OpenSim or set by the users, are further transformed into spike trains by the **Motor Unit Pool** model. Surface EMG signals are finally calculated as the summation of the convolution of the dynamic MUAPs and the spike trains among all active motor units.

MSK model used in NeuroMotion. The original ARMs model includes 23 degrees of freedom (DoFs), including the full DoFs of finger movements and the flexion/extension and radial/ulnar deviation DoFs for the wrist. We added the pronation/supination DoF to the source files of ARMs model such that a full range of hand, wrist, and forearm movements could be simulated.

The MSK model in NeuroMotion can be customised to the subject's anthropometry in three ways. First, a fully personalised MSK model can be created by acquiring medical images of the MSK system and building physics-based models. This process requires considerable skills and manual interventions, but has been facilitated by the advancement of automatic tools [18, 19] and associated machine learning methods [20]. A more practical approach, which is automated in OpenSim, is to scale each segment in the ARMs model by minimising the distances between the virtual markers placed on ARMs model and the real markers placed on the subject. The ARMs model can also be manually scaled by anthropometric measurements.

The movement defined by the user can be visualised and investigated in the OpenSim GUI. Muscle fibre lengths during the movement are related to the joint angles and are tracked and extracted using the OpenSim built-in functions. Assuming tendons to be rigid at constant slack length, the muscle fibre lengths are deduced and imported into BioMime to generate the dynamic MUAPs. Furthermore, the individual muscle activations responsible for the movement are estimated with the built-in Static Optimization tool in OpenSim. Static Optimisation solves the inverse dynamics by minimising the cost function of muscle activations under the constraints of muscle activation-to-force conditions. The exported muscle activations are further normalised and imported into the Motor Unit Pool model to act as the neural input to the motoneuron pools, according to the linearity properties of the motoneuron pools [21]. OpenSim's built-in tools have been integrated into NeuroMotion by its python API.

BioMime

After OpenSim captures the changes in a biomechanical system from the joint kinematics, BioMime is used to simulate the corresponding MUAPs during the movement. BioMime is a deep conditional generative model that simulates electrical potential fields given a set of physiological parameters [16]. A schematic of BioMime's training and inference pipeline is shown in Figure 2. Learning from the outputs of its teacher numerical model [10], BioMime captures the relations between the physiological parameters and the MUAP templates and essentially replicates the biophysical properties of the volume conductor of the forearm. When the time-varying parameters during a movement are imported into BioMime, the output will change accordingly. Therefore, BioMime can be used to generate a sequence of MUAP templates during any movement as long as the parameter changes are available and plausible. The computational cost to simulate a movement in high temporal resolution is extremely low given BioMime's ultra-fast inference speed (0.287 seconds per muscle per condition [16]).

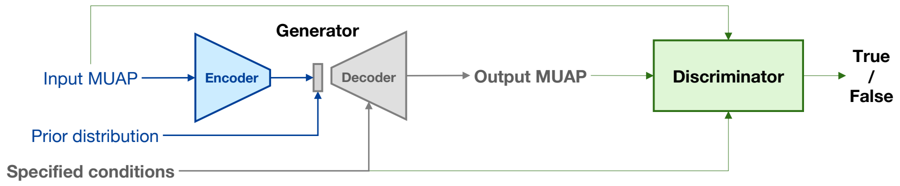

Fig 2. Schematic of BioMime. BioMime is a conditional generative model trained in an adversarial way. The generator takes the specified conditions and latent features of MUAPs as the inputs and outputs the simulated MUAP signals. The latent features can be encoded from existing MUAPs or sampled from a prior distribution. The discriminator distinguishes real samples from fake samples conditioned on the specified parameters. Only the generator is used in NeuroMotion.

The public BioMime model was trained on a dataset generated by a numerical model with a specific forearm anatomy. Therefore, the BioMime model can be regarded as a subject-specific surrogate model of the volume conductor with this forearm anatomy. The user can keep this well-trained model or train their individual BioMime. Detailed instructions on how to train BioMime can be found in https://github.com/shihan-ma/BioMime. The newest version of BioMime supports changes in seven physiological parameters, including the conductivity of the fat layer and the fibre number, depth and medial-lateral position, innervation zone, conduction velocity, and fibre length of each motor unit (Details in the supplementary of [16]). Some of these parameters can be measured experimentally. For example, the location and the fibre length of a motor unit could be estimated by using ultrasound [22, 23]. The location of the innervation zone and the muscle fibre conduction velocity can be estimated, for example by using high-density EMG recordings [24, 25]. Some parameters can also be obtained from the synthetic movement of the MSK model. As described in Section MSK Model with OpenSim, muscle fibre length can be estimated by OpenSim. The depth of a motor unit territory and the motor unit conduction velocity can be approximated from the muscle fibre length under some assumptions (see Section Estimate Parameter Changes).

As a conditional generative model trained in an adversarial manner, BioMime has an encoder-decoder structured generator and a discriminator. During the inference, only the generator is required. BioMime can be used to generate new MUAPs by morphing an existing MUAP or by sampling from a prior distribution. The difference is that by morphing an existing MUAP, the output MUAPs keep the properties of the input MUAP that are separated from the seven physiological parameters. These properties are sampled from a prior distribution by sampling. Therefore, generating MUAPs by sampling only uses the decoder. Since the seven parameters explain the bulk of variations of the MUAP templates of the specific subject, the difference between the MUAPs generated by morphing or by sampling is small. NeuroMotion supports both of these two strategies and provides a dataset of MUAPs paired with their physiological parameters for generation under the morphing pattern.

Motor Unit Pool

The purpose of including the motor unit pool model in NeuroMotion is twofold: (1) to initialise the properties of the motor units within one muscle, and (2) to simulate the motoneuron activities and generate their spike trains. In NeuroMotion, we implemented two types of motor unit pool models, the classical Fuglevand's model proposed in [14] and a cohort of leaky fire-and-integrate (LIF) neuron models adapted from [26].

The properties of the motor units in the two models are initialised following the rules below. In both models, the motor unit size (number of muscle fibres within each unit) is proportional to the amplitude of the motor unit twitch force [14, 27], and the summation of the motor unit sizes meets the expected total number of fibres in one muscle (Equation 1). In the classical model, the peak twitch force is exponentially distributed while in the LIF-based model, the twitches are distributed following the linear-exponential function in Equation 2. Equation 2 was derived from experimental measurements in the literature on human forearm muscles [14, 28, 29] following the method described in [26]. The total number of fibres in one muscle was estimated as the ratio between muscle physiological cross-sectional area and the average fibre cross-sectional area for typical human forearm muscles [14, 30]. In the classical Fuglevand's model, the conduction velocity is normally distributed within the common range and then sorted from small to large to be positively related to the motor unit size. In the LIF-based model, we adapted the mathematical relationships between the axonal conduction velocity and the neuron surface area from [31] to estimate the fibre conduction velocity. In both models, the depth and medial-lateral positions of motor units are uniformly distributed within the muscle territories. The innervation zone and fibre length are normally distributed within the common ranges.

$$MN_{size} = \frac{\overline{f^{tw}}(j)}{\sum_{k=1}^{N} \overline{f^{tw}}(k)} \cdot N_f, j \in [1, N] \tag{1}$$

where $\overline{f^{tw}}(j)$ is the estimated twitch force of the jth motor unit in the population, N the number of motor units, and N_f the total number of fibres in one muscle.

$$\overline{f^{tw}}(j) = 0.81 \cdot (18.51 \cdot \frac{j}{N} + 104.10^{\frac{j}{N}^{4.83}}), j \in [1, N] \tag{2}$$

In both models, the motoneurons innervating a muscle are sequentially recruited in terms of the intensity of the neural input. The smallest motor unit is first recruited with the lowest impulse response amplitudes. Once recruited, the motoneuron starts to fire regularly. The firing rates of the recruited motor units increase with the input to the motoneuron pool until the peak firing rates are reached. The two models both predict the specific discharge behaviour (spike trains) of the motoneurons from the intensity of the neural input. Specifically, the firing state of a motor unit is decided by the recruitment threshold of the motor unit in the classical Fuglevand's model [14]. The recruitment thresholds are exponentially distributed such that few motor units have high threshold and many motor units have low threshold. In the LIF model-based approach, each LIF model captures the complex nonlinear MN dynamics by using a parallel combination of a leaky resistor and a capacitor. A spike is generated when the transmembrane voltage reaches a certain threshold, followed by a refractory during the action potential. With physiologically realistic distributions of the electrophysiological properties and a further interpolation between the motoneuron-specific electrophysiological properties supported by experimental data [26] in the LIF model, the predictions of both models are consistent with the onion skin theory [32] and Henneman's size principle of sequential motoneuron recruitment [33].

Toolbox Functions

In this section, we introduce the basic functions provided by NeuroMotion to define the movement of the ARMs model (Section Define Movement), track the changes in physiological parameters (Section Estimate Parameter Changes), set common drives to motoneuron pools (Section Set Neural Inputs to Motoneuron Pools), and define the structure of motor unit pools (Section Configure Motor Unit Pools). Python code examples for each function are displayed.

Define Movement

Movements are simulated by using the ARMs model in NeuroMotion. A Python class ***MSKModel*** was implemented to handle the related functions. NeuroMotion supports three approaches to defining the movement of the ARMs model. The most basic way is by assigning joint angles to each DoF by using the function *'load_mov'* (Figure 3a). The joint angles can be obtained from motion capture data (like in [34]) or be measured from sensors (as by data glove[1] and angle sensors). There are 24 DoFs in the ARMs model and each can be changed separately within a predefined range. NeuroMotion automatically checks whether each input joint angle is within the correct range. It is possible that a movement defined by a set of joint angles is not feasible even though each joint angle is reasonably assigned. For example, a combination of joint angles may result in a sudden and unlikely change in muscle fibre lengths. We encourage the users to visualise the movement in OpenSim GUI and to check if there are such aberrations. The motion file required by OpenSim can be generated from joint angles by using the function *'write_mov'* in NeuroMotion (Python code example at Line 10 in Code Block 1).

The second way to define a movement is by interpolating between predefined poses. NeuroMotion provides eight default poses, including hand open/grasp, wrist flexion/extension, wrist radial/ulnar deviation, and forearm pronation/supination (Figure 3b). Users can define their customised poses by creating new poses in OpenSim GUI. A movement can be simulated by setting one pose at each stage from the predefined poses and setting the durations of each transition (Python code example at Lines 1-5 in Code Block 1). The eight default poses can be combined before interpolation by concatenating the names of the poses, e.g., 'open+flex' means hand open and wrist flexion at the same time. The joint angles are summed after the combination. Users may want to define the movement directly from the motion capture data, as the third way displayed in Figure 3c. This is possible with OpenSim GUI. The users need to place virtual markers at the same anatomical positions on the MSK model as the experimental positions. Then the MSK model can be scaled and the movement can be driven by the motion capture data.

```
1    # build msk model and simulate a movement
```

[1]https://5dt.com/5dt-data-glove-ultra/

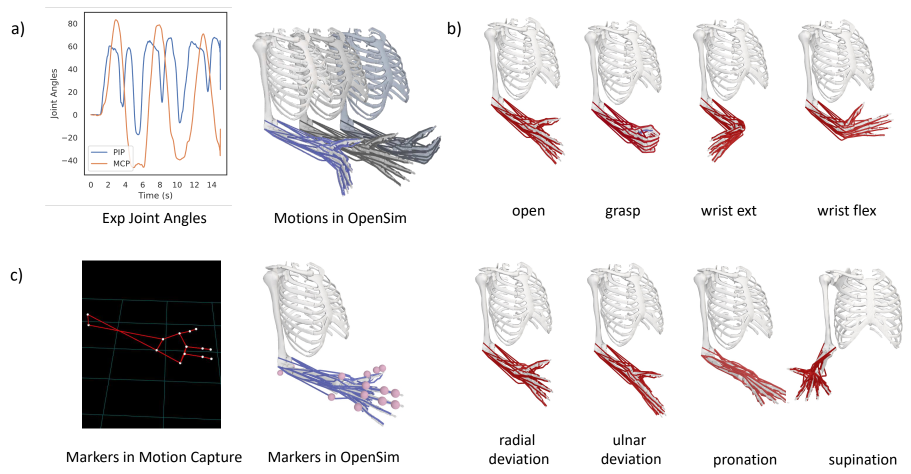

Fig 3. Illustration of the function of movement definition in NeuroMotion. The MSK models are driven by the joint angles in OpenSim. NeuroMotion provides three ways to define a movement. a), The most basic way to define a movement is by setting the joint angles directly. In this example, the ARMs model is driven by the MCP (metacarpophalangeal) and PIP (proximal interphalangeal) joints of the five digits. b), Eight default poses are provided. Users can combine these poses and interpolate between them to simulate a smooth movement. c), Motion capture data can be used to drive the MSK model. Markers should be aligned between experimental and simulation settings.

```
2     msk = MSKModel()
3     poses = ['default', 'default+flex', 'default']
4     durations = [2] * 2    # 2 seconds for each transition
5     fs = 5    # 5 Hz
6     ms_labels = ['ECRB', 'ECRL', 'PL', 'FCU', 'ECU', 'EDCI', 'FDSI']
7     msk.sim_mov(fs, poses, durations)
8     ms_lens = msk.mov2len(ms_labels=ms_labels)
9     changes = msk.len2params()
10    msk.write_mov('./res/mov.mot')
```

Code Block 1. Example Python codes for defining a wrist flexion movement by interpolation and for acquiring changes in physiological parameters.

Estimate Parameter Changes

BioMime can simulate MUAPs during the changes in seven physiological parameters. Users can import the parameters measured during an experiment. If these physiological parameters are not available from the experiment, NeuroMotion provides an alternative method to approximate some of the parameters. Muscle fibre lengths during a movement of the MSK model can be estimated by using the Muscle Analysis Tool in OpenSim. Two more parameters can be approximated from the muscle fibre length changes. Under the assumption of constant muscle volumes during a movement [11], the cross-sectional area of a muscle changes inversely with the muscle fibre length. Then the conduction velocity can be tracked since it is positively correlated with the cross-sectional area of muscle volumes. If the location of a motor unit consistently changes with the radius of the cross-section, the depth of the motor unit can be estimated as well. Therefore, changes in three physiological parameters can be estimated during a movement. These changes can be obtained by using the function of *'len2params'* in NeuroMotion (Python code example at Line 9 in Code Block 1).

Set Neural Inputs to Motoneuron Pools

Activations that trigger muscle contractions are often represented by spike trains. The spike trains are generated by motoneurons by transforming the input they receive and are convolved with the MUAP templates to generate the interference EMG signals. It still remains a challenge to decode the real neural input to motoneurons from experimental EMG data, and thus it is difficult to set these inputs to their true values during a movement.

NeuroMotion provides three ways to set the input to each motoneuron pool. First, the user can set predefined and commonly used activation profiles, which include constant, trapezoidal, triangular, and sinusoidal activations. The amplitude and duration of each type of activation can be easily changed. Second, as described in Section MSK Model with OpenSim, muscle activations during a movement can be estimated by OpenSim with the Static Optimisation Tool. The muscle activations are then normalised to the activations during the maximum voluntary contractions and used as the neural inputs to the muscles. Third, NeuroMotion provides the interfaces for the user to set the muscle activations by the outputs of their algorithms, for example, using the normalised EMG signals recorded during experiments or the signals derived from muscle synergies [35].

Configure Motor Unit Pools

NeuroMotion organises the motor unit pool structures by providing a Python class ***MotoneuronPool***. This class parameterises the electrical and mechanical properties of a motor unit pool, of which each property can be easily customised. Fundamental properties include the motor unit size, conduction velocity, motor unit positions, innervation zones, and fibre lengths. In the implementation of classical Fuglevand's model, the motor unit recruitment threshold, the minimum and maximum firing rate of each motor unit, variability of the inter-pulse intervals, and motor unit twitch force and contraction time can be manually assigned. In the LIF-based model, the motoneuron-specific parameters, such as the resistance, capacity, and time constant, are set given the mathematical descriptions in [26]. Example Python codes that define a classical motor unit pool model, define the input to motoneurons, and calculate the spike trains are shown in Code Block 2.

```
# test motor unit pool model
mn_pool = MotoneuronPool(num_mu, rr, rm, rp, pfr1, pfrd, mfr1, mfrd,
ge, c_ipi, frs1, frsd)         # parameters that define the motor unit
 recruitment and firing pattern
# properties
config = edict({
    'num_fb': 25000,
    'depth': [20, 30],
    'angle': [20, 30],
    'iz': [0.5, 0.1],
    'len': [0.5, 0.1],
    'cv': [4, 0.5]
})
properties = mn_pool.assign_properties(config, True)
# define input drive
fs = 2048          # Hz
duration = 6       # s
ext = np.linspace(0, 1.0, fs * duration)
# Force and twitches
mn_pool.init_twitches(fs)
mn_pool.init_quisistatic_ef_model()
# spike trains
```

```
    _, spikes, fr, ipis = mn_pool.generate_spike_trains(ext)
```

Code Block 2. Example Python codes for defining a motor unit pool model and for setting the input drive to the pool.

Results

NeuroMotion provides the input, output, and all intermediate variables to the users, including the joint kinematics and the muscle fibre lengths during a movement, the neural input to each motoneuron pool, the neural commands to the muscles in the format of spike trains, the dynamically changing MUAPs, and the interference surface EMG signals. Given the above information, NeuroMotion can be used to analyse the impact of a specific parameter on the EMG signals and to provide synthetic datasets for data augmentation and validations of EMG-related algorithms. Here we displayed the variations of MUAPs during dynamic contractions and compared the similarities between MUAPs within and across muscles. A case study was provided in which the synthetic dataset generated by NeuroMotion was used to augment an experimental dataset for improving the accuracy in estimating joint angles from EMG signals.

Synthetic MUAPs during Dynamic Contractions

MUAPs When One, Two, and Three Parameters Change

NeuroMotion allows researchers to estimate the changes in muscle fibre length, conduction velocity, and motor unit depth during the movement of the ARMs model (Sections MSK Model with OpenSim and Estimate Parameter Changes). Here, we gave examples of the simulated MUAPs during a wrist flexion and extension movement (Figure 4a) with four levels of parameter changes, including changing only muscle fibre length, changing muscle fibre length and conduction velocity, changing muscle fibre length and motor unit depth, and changing all three parameters. One representative MUAP from the ulnar head of Flexor carpi ulnaris (FCU(u)) and its changes over time are displayed in Figure 4b. By changing the muscle fibre length, the duration of the MUAPs slightly varied. Changing the conduction velocity also influenced the duration of the MUAP while changing the motor unit depth had a more substantial impact on the waveforms. Changes due to the three physiological parameters during the movement are visualised in Figure 4c, d, and e, respectively.

MUAPs across Poses

We then proceeded to study how MUAPs from different muscles change across movements. Three basic movements were simulated by using the tools described in Section Define Movement, including sequences of hand-open to hand-grasp, hand-open to wrist flexion and extension, and hand-open to radial and ulnar deviations. We visualised the changes in MUAPs from six superficial muscles. As shown in Figure 5, only the lengths of digitorum muscles (extensor digitorum, ED and flexor digitorum superficialis, FDS) changed during hand grasp and open movement. As a result, MUAPs from ED and FDS displayed large variations while MUAPs from the other muscles maintained their waveforms. All six muscles were activated during the flexion and extension movement. Therefore, the duration, amplitude, and waveform of the MUAPs from the six muscles all changed consistently with the movement. The digitorum muscles (ED and FDS) and palmaris longus (PL) muscle contributed less to the radial and ulnar deviations, and thus MUAPs from these three muscles showed fewer variations during wrist deviations.

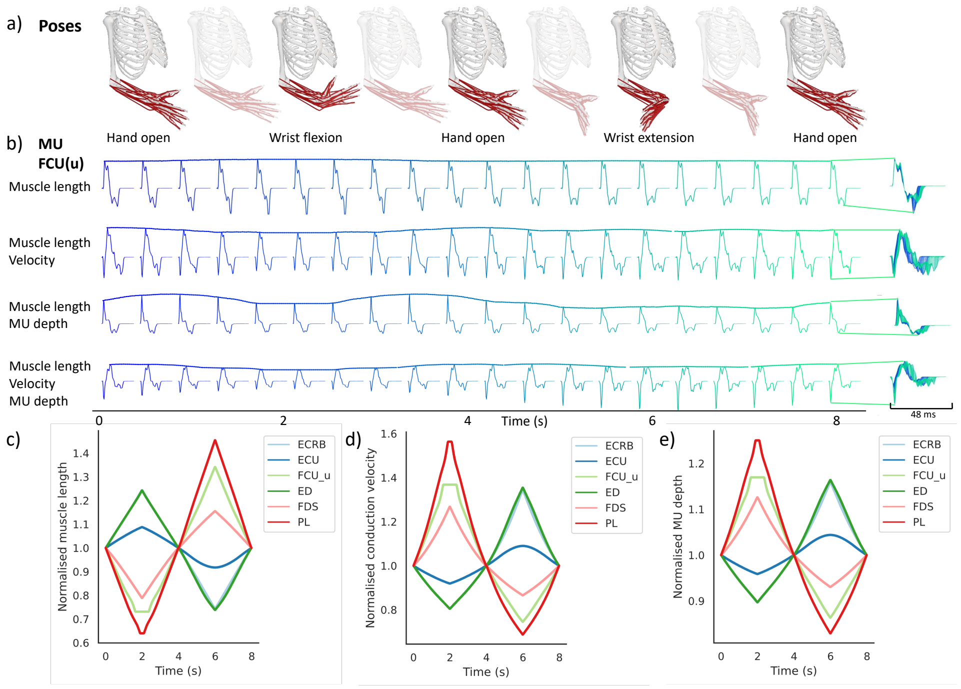

Fig 4. Changes in the MUAPs of one representative motor unit in FCU(u) during a wrist flexion/extension movement. a), Sequence of the movement. b), MUAPs generated by NeuroMotion. The MUAPs in the first row were morphed using the muscle fibre length profiles output by the ARMs model using OpenSim, the second row using muscle fibre length and conduction velocity, the third row using muscle fibre length and motor unit depth, and the fourth row using all three parameters. c), Normalised muscle fibre length profiles from the ARMs model. d), Normalised conduction velocity. e), Normalised motor unit depth.

In-muscle and Cross-muscle MUAP Similarities

Muscles within the forearm have small volumes. Thus, MUAPs within a single forearm muscle might have similar waveforms due to the overlapping motor unit territories. We studied the similarities between MUAPs within a single muscle, across pairs of muscles, and across joint angles during a wrist flexion and extension movement. The similarity was evaluated by the normalised mean square error, which is defined by the mean square error between two MUAPs divided by their averaged power. The MUAPs were first cropped to a sub-grid where all channels had the maximum amplitude above 75% of the average across channels.

The similarities between MUAPs of three muscles, an extensor (Extensor carpi radialis brevis, ECRB) and two heads of one flexor (Flexor carpi ulnaris ulnar and humeral heads, FCU(u) and FCU(h)), are shown by the confusion matrices in Figure 6. MUAPs within the same muscle showed higher similarity than MUAPs across different muscles. MUAPs within the two heads of the FCU muscle were more similar than MUAPs within the FCU and the ECRB muscles.

We were also interested in how movements influence the similarities of the MUAPs and whether the changes in the similarities during the movement are consistent across MUAPs within each muscle. Figure 7 shows the similarities between MUAPs morphed during a wrist flexion/extension movement and their baseline shapes (MUAPs at zero joint angles). MUAPs were gradually morphed away from their baseline shapes when

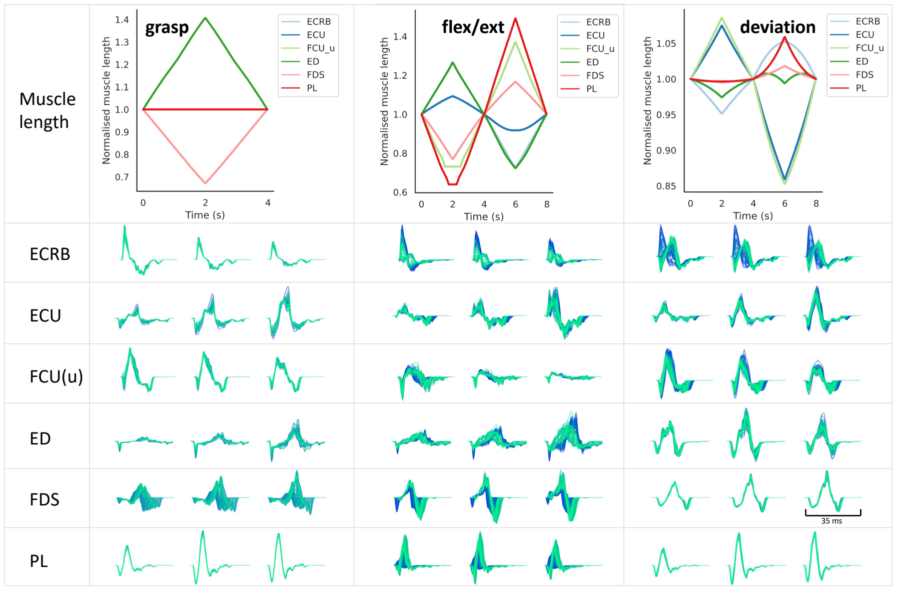

Fig 5. Changes in MUAPs during three movements. One representative MUAP was chosen from each of the six muscles and was consistently morphed during three movements. The first row: normalised muscle fibre length profiles from the ARMs model during grasp-open, flexion-extension, and radial-ulnar deviation. The second to the seventh rows: MUAPs transformed during the three movements in the six muscles. The colours from dark blue to light green indicate the time frames from the beginning to the end of the movement. For each MUAP, a subgrid of three channels were chosen for display, which was consistent across the three movements.

the joint flexion or extension angle increased. The similarities of MUAPs across joint angles were consistent within the MUAPs of each muscle with small variances. MUAPs in extensors showed higher levels of dissimilarity during wrist extension than during wrist flexion, and the opposite for MUAPs in flexor muscles.

Synthetic EMG signals for data augmentation

The synthetic EMG signals generated by NeuroMotion can be used to augment experimental datasets for myoelectric control. In this case study, we simulated hand and wrist movement following the experiments in [36]. In the experiments, each subject performed simultaneous wrist and metacarpophalangeal (MCP) flexion/extension with self-paced speed in five trials. Single-channel EMG signals on six forearm flexors and extensors were recorded, as well as the wrist and MCP joint angles. The EMG signals were normalised by the maximum EMG signals during maximum voluntary contractions and then used as the neural inputs to the corresponding muscles to produce spike trains. The joint angles were used to drive the ARMs model, where the output muscle fibre lengths were imported to BioMime to simulate dynamic MUAPs. Finally, the synthetic EMG signals were simulated by convolving the spike trains and the MUAPs.

We simulated the synthetic EMG data from three subjects. Six out of 320 channels were selected from NeuroMotion's output by matching the positions of the real electrodes and the simulated electrodes. EMG signals during maximum voluntary contractions were also simulated to normalise the synthetic data. Root mean square values were computed from 200-ms intervals with 50-ms overlapping. Each sample was labelled with the two joint angles.

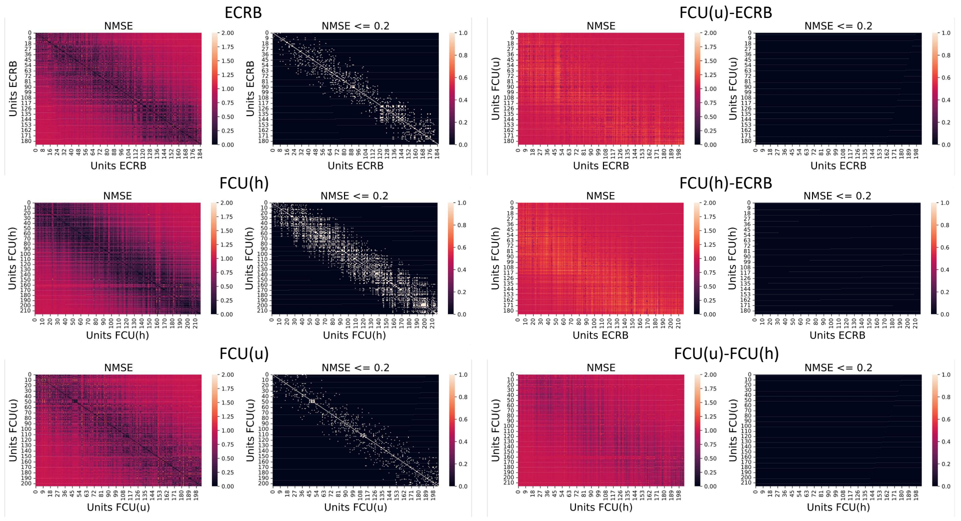

Fig 6. Similarity between MUAPs within/across muscles in confusion matrices. Deeper colours in the first and the third columns indicate a lower normalised mean square error (NMSE) and higher similarity. Black denotes NMSE > 0.2 (less similar) and white NMSE < 0.2 (more similar) in the second and the fourth columns. The left two columns show similarity between MUAPs within single muscles while the right two columns show similarity between MUAPs across two muscles.

We show that the synthetic EMG signals are of practical use in two ways. First, we trained ridge regressors for each subject and each DoF on the synthetic dataset. The trained regressors were directly applied to regress the joint angles from the experimental EMG signals of the same subject. The averaged Pearson correlation coefficients for one subject were 0.54 and 0.72 for wrist and MCP, respectively. For the other two subjects, the best regressors predicted the two joint angles with Pearson correlation coefficient $>$ 0.5 in one trial. An example of regression results is shown in Figure 8 a.

Second, we show that the synthetic data can be used to augment the experimental data. Specifically, we randomly selected two trials of experimental data and two trials of synthetic data to form a new training dataset. The ridge regressors trained on this augmented dataset were tested on one trial of the experimental data, which was different from the trials in the training dataset. For two subjects, the performance of the regressor was improved when trained on the augmented dataset. For example, as shown in Figure 8 b, the Pearson correlation coefficient of regressing wrist improved from 0.64 to 0.71 and MCP from 0.54 to 0.57.

Discussion

We have presented NeuroMotion, which provides the first open-source simulator for neuromechanical investigations by modelling the electrical outputs during voluntary human movements. With the three key modules, NeuroMotion provides a wealth of functions that give full freedom to users to simulate signals by using the default settings or by customising the configurations. All input, intermediate, and output variables are available in this hierarchical and full-spectrum simulation, indicating multiple potential usages of NeuroMotion.

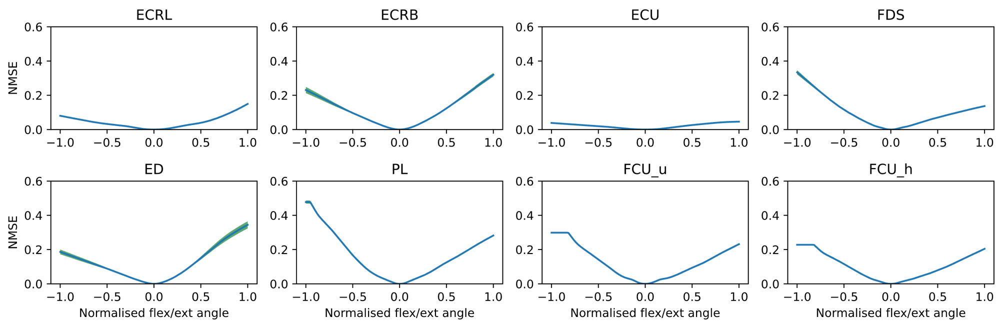

Fig 7. Similarity between MUAPs across joint angles during a wrist flexion/extension movement. The joint angles were normalised between -1.0 (flexion) and 1.0 (extension). MUAPs at zero flexion/extension were used as the baseline shapes, which were compared with the continuously morphed MUAPs during the movement to calculate the NMSE (dissimilarity metric, higher values indicate higher differences). NMSEs were averaged within each muscle at each time step, with the variances shown by the green shaded area.

Rich Variances of MUAPs during Dynamic Contractions

NeuroMotion provides three ways to define the movement of the ARMs model. The easiest way is to interpolate between (the combinations of) the eight default poses. Examples in Figure 4 and Figure 5 showed simulations of MUAPs by using the interpolation tools. Physiological parameters were smoothly changed during the movement, which resulted in a continuous variation of MUAP waveforms. The variations of synthetic MUAPs are similar to the variations of MUAPs observed during experiments in a qualitative way [37–39]. For example, the shortening of muscle fibres or increasing of conduction velocity during natural muscle contractions reduced the duration of a MUAP; MUAPs of muscles that contribute less to a movement showed fewer variations (PL in wrist deviation). Compared with the limited muscles studied under a few discretised joint angles during experiments, NeuroMotion can simulate MUAPs from eight forearm superficial muscles under any voluntary movements in a high temporal resolution. Such functionality allows users to test the assumptions made in the simulations (e.g., changes in parameters) or study the rich variances of MUAPs within or across muscles as displayed in Figure 6. The ability of NeuroMotion to generate dynamic MUAPs consistent with muscle functionalities demonstrates its superiority to previous models that either adapted the classical cylindrical model to discretised stages [40] or empirically transformed the MUAP waveforms [41]. We expect that NeuroMotion will meet the demand for providing synthetic EMG signals during dynamic muscle contractions for validating adaptive decomposition algorithms [37].

We also showed that the generated MUAPs were muscle-specific with low similarity across different muscles (Figure 6). The variations of MUAPs during a movement were consistent within muscle but different across muscles (Figure 7). When the wrist flexion/extension angle increased, MUAPs were gradually transformed and deviated away from their initial version, leading to an increasing difference (higher NMSE). The similarity at each joint angle showed small variances across MUAPs within each muscle, which indicates that deformations of fibres within one muscle are similar. MUAPs from flexors experienced more variations during flexions while MUAPs from extensors showed more variations during extensions. This observation fits well with the fact that flexors/extensors manifest a more drastic contract during flexions/extensions. These results evidence the potential of using NeuroMotion to investigate the variations of MUAPs within/across muscles during voluntary movements.

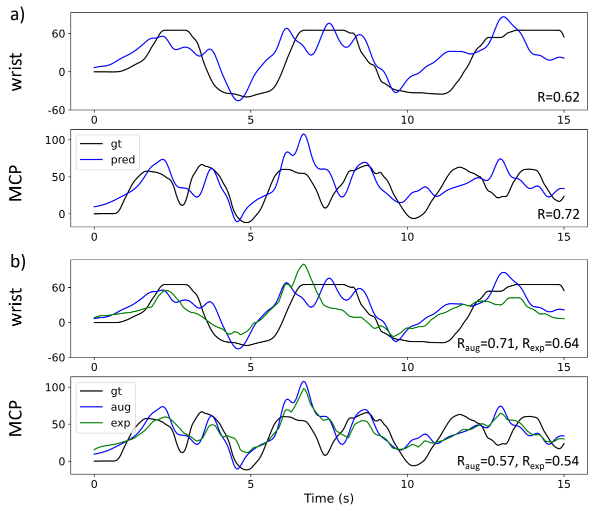

Fig 8. Representative results of using NeuroMotion's output to train regression algorithms and to augment the experimental dataset. a), The regressors trained on the synthetic dataset could predict wrist and MCP joint angles from the experimental data, with the Pearson correlation coefficient (R) 0.62 and 0.72, respectively. b), When trained on the dataset augmented by the synthetic signals, the ridge regressors achieved better performance compared with the regressors trained only on the experimental data.

Test-bed for Cause-Effect Studies and Data Augmentation for Regression Analysis

With access to all inputs, outputs, and intermediate variables in NeuroMotion, users can study cause-effect relationships between the joint kinematics, neural drives, physiological parameters, muscle forces, MUAPs, spike trains, and surface EMG signals. As a simple demonstration of such studies, we showed how MUAP changes differently with one to three physiological parameters changed during a wrist flexion/extension movement (Figure 4). Coupled with experimental MUAPs observed at the same muscle under discretised movements, such simulations will help test the assumptions made in Section Estimate Parameter Changes and elucidate the contributions of the physiological parameters to the MUAPs. Another example displays the transformations of MUAPs under three wrist-and-hand movements (Figure 5). This allows the investigation of how MUAP changes when the commonly used forearm DoFs are activated, which will further provide insights into designing and improving current adaptive decomposition algorithms.

NeuroMotion can also be used to study the "inverse" relations between the outputs and the inputs of the system, for example, to pre-train algorithms that regress joint angles from EMG signals and to augment the experimental dataset. Here we gave an example to show that ridge regressors trained on the synthetic dataset can be directly applied to regress joint angles from experimental signals. Furthermore, we found that the performance of the regressors could be improved when trained on the augmented dataset. These findings are consistent with the perspective of using synthetic datasets to augment the limited actual dataset and facilitate algorithm training [10, 42]. Another important observation from the regression example is that the properties of the synthetic EMG signals largely depend on neural inputs/spike trains. It still remains a major challenge to decode the real neural inputs during a voluntary movement, and thus impossible to set an exactly accurate neural input in NeuroMotion. Here, we used the normalised EMG signals recorded on each muscle as the neural inputs. The flexibility provided by NeuroMotion allows the investigation of the actual relationship between the neural input and the joint kinematics, for instance, by adjusting the input drives to make the synthetic EMG signals more similar to the experimental data under the same protocols. This will in turn give insights into the real relations between joint kinematics and neural inputs during voluntary movements.

Limitations and Future Directions

As the first integrated open-source generative model to simulate EMG signals during voluntary movements, NeuroMotion still has some limitations that should be noted. First, the current BioMime model in NeuroMotion is used as a replica of a specific volume conductor system but not a universal one. A related limitation is that the locations of the electrodes are fixed (10×32 channels around the forearm) and cannot be specified. This can be addressed by training a subject-specific BioMime with different electrode configurations. Details of data preparation and model training can be found in [16]. Building a universal BioMime model that can be adapted to specific subjects and allows a random sampling of electrical potentials on the surface is also in our future work.

Second, the muscle-related results obtained from MSK simulations are highly sensitive to some MSK parameters [43–46], including the muscle-specific maximum isometric force, tendon slack length, optimal fibre length, and the muscle moment arm. These parameters in the ARMs model may deviate from the specific subject, which will affect the estimation of muscle activation. Scaling the ARMs model or building a personalised MSK model can address some of the above issues. The personalised models could be further used to simulate patient's data, such as fatigue and limited mobility in people with spinal cord injuries.

Third, changes in physiological parameters during voluntary movement are estimated by first extracting the changes in muscle fibre length using OpenSim and then approximating the changes in conduction velocity and motor unit depth under physiological assumptions. Currently, the fibre lengths are derived from the geometrical muscle-tendon lengths computed with the ARMs model in OpenSim by assuming the tendons to be rigid and at constant slack length [27, 47]. This assumption of stiff tendon [48] is acceptable given that the investigated muscles have ratios of tendon slack length to optimal fibre length below 5.0 [48]. However, the muscle-tendon interacting dynamics during contraction are not considered under this assumption, which might introduce errors in estimating the changes in muscle fibre length and muscle activation. We expect that the accuracy of NeuroMotion will be improved with the advancement of *in vivo* measurements of physiological parameters during movements.

Conclusion

We proposed NeuroMotion, an open-source EMG simulator that generates surface EMG signals during human forearm movements. For the first time, NeuroMotion provides a core resource for the neuromechanics community to address the problem of simulating physiological electrical outputs during biomechanical movements and allows a full-spectrum simulation from movements to neural commands and EMG signals. All intermediate variables (kinematics, dynamic MUAPs, spike trains) are available during the simulation, which makes the model flexible to use for the users' own purposes. We demonstrated that one potential application is to use synthetic data to augment the experimental data for training regression algorithms. We would expect that with the functionality and extensibility provided by NeuroMotion, users can customise the way they utilise NeuroMotion that will ultimately yield progress in the neuromechanics fields.

References

1. Valero-Cuevas FJ. Fundamentals of Neuromechanics. vol. 8. Springer; 2016.
2. Ting LH, McKay JL. Neuromechanics of muscle synergies for posture and movement. Current Opinion in Neurobiology. 2007;17(6):622–628.
3. Schwartz AB. Movement: how the brain communicates with the world. Cell. 2016;164(6):1122–1135.
4. Nishikawa K, Biewener AA, Aerts P, Ahn AN, Chiel HJ, Daley MA, et al. Neuromechanics: an integrative approach for understanding motor control. Integrative and Comparative Biology. 2007;47(1):16–54.
5. Ting LH, Chiel HJ, Trumbower RD, Allen JL, McKay JL, Hackney ME, et al. Neuromechanical principles underlying movement modularity and their implications for rehabilitation. Neuron. 2015;86(1):38–54.
6. Farina D, Jiang N, Rehbaum H, Holobar A, Graimann B, Dietl H, et al. The extraction of neural information from the surface EMG for the control of upper-limb prostheses: emerging avenues and challenges. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2014;22(4):797–809.
7. Hodgkin AL, Huxley AF. A quantitative description of membrane current and its application to conduction and excitation in nerve. The Journal of Physiology. 1952;117(4):500.
8. Farina D, Rainoldi A. Compensation of the effect of sub-cutaneous tissue layers on surface EMG: a simulation study. Medical Engineering & Physics. 1999;21(6-7):487–497.
9. Farina D, Mesin L, Martina S, Merletti R. A surface EMG generation model with multilayer cylindrical description of the volume conductor. IEEE Transactions on Biomedical Engineering. 2004;51(3):415–426.
10. Maksymenko K, Clarke AK, Mendez Guerra I, Deslauriers-Gauthier S, Farina D. A myoelectric digital twin for fast and realistic modelling in deep learning. Nature Communications. 2023;14(1):1600.
11. Mesin L, Joubert M, Hanekom T, Merletti R, Farina D. A finite element model for describing the effect of muscle shortening on surface EMG. IEEE Transactions on Biomedical Engineering. 2006;53(4):593–600.
12. Delp SL, Anderson FC, Arnold AS, Loan P, Habib A, John CT, et al. OpenSim: open-source software to create and analyze dynamic simulations of movement. IEEE Transactions on Biomedical Engineering. 2007;54(11):1940–1950.
13. Seth A, Hicks JL, Uchida TK, Habib A, Dembia CL, Dunne JJ, et al. OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. PLoS Computational Biology. 2018;14(7):e1006223.
14. Fuglevand AJ, Winter DA, Patla AE. Models of recruitment and rate coding organization in motor-unit pools. Journal of Neurophysiology. 1993;70(6):2470–2488.
15. Schmidt MD, Glasmachers T, Iossifidis I. The concepts of muscle activity generation driven by upper limb kinematics. BioMedical Engineering OnLine. 2023;22(1):1–29.

16. Ma S, Clarke AK, Maksymenko K, Deslauriers-Gauthier S, Sheng X, Zhu X, et al. Conditional Generative Models for Simulation of EMG During Naturalistic Movements. arXiv preprint arXiv:221101856. 2023;.

17. McFarland DC, Binder-Markey BI, Nichols JA, Wohlman SJ, de Bruin M, Murray WM. A Musculoskeletal Model of the Hand and Wrist Capable of Simulating Functional Tasks. IEEE Transactions on Biomedical Engineering. 2022;.

18. Modenese L, Renault JB. Automatic generation of personalised skeletal models of the lower limb from three-dimensional bone geometries. Journal of Biomechanics. 2021;116:110186.

19. Valente G, Crimi G, Vanella N, Schileo E, Taddei F. nmsBuilder: Freeware to create subject-specific musculoskeletal models for OpenSim. Computer Methods and Programs in Biomedicine. 2017;152:85–92.

20. Saxby DJ, Killen BA, Pizzolato C, Carty C, Diamond L, Modenese L, et al. Machine learning methods to support personalized neuromusculoskeletal modelling. Biomechanics and Modeling in Mechanobiology. 2020;19:1169–1185.

21. Farina D, Negro F. Common synaptic input to motor neurons, motor unit synchronization, and force control. Exercise and Sport Sciences Reviews. 2015;43(1):23–33.

22. Maitland S, Hall J, McNeill A, Stenberg B, Schofield I, Whittaker R. Ultrasound-guided motor unit scanning electromyography. Muscle & Nerve. 2022;66(6):730–735.

23. Hauraix H, De Monsabert BG, Herbaut A, Berton E, Vigouroux L. Force-length relationship modeling of wrist and finger flexor muscles. Medicine and Science in Sports and Exercise. 2018;50(11):2311–2321.

24. Zhang C, Dias N, He J, Zhou P, Li S, Zhang Y. Global innervation zone identification with high-density surface electromyography. IEEE Transactions on Biomedical Engineering. 2019;67(3):718–725.

25. Martinez-Valdes E, Farina D, Negro F, Del Vecchio A, Falla D. Early motor unit conduction velocity changes to high-intensity interval training versus continuous training. Medicine & Science in Sports & Exercise. 2018;50(11):2339–2350.

26. Caillet AH, Phillips AT, Farina D, Modenese L. Estimation of the firing behaviour of a complete motoneuron pool by combining electromyography signal decomposition and realistic motoneuron modelling. PLOS Computational Biology. 2022;18(9):e1010556.

27. Caillet AH, Phillips AT, Farina D, Modenese L. Motoneuron-driven computational muscle modelling with motor unit resolution and subject-specific musculoskeletal anatomy. bioRxiv. 2023; p. 2023–06.

28. Romaiguère P, Vedel JP, Pagni S, Zenatti A. Physiological properties of the motor units of the wrist extensor muscles in man. Experimental Brain Research. 1989;78:51–61.

29. Riek S, Bawa P. Recruitment of motor units in human forearm extensors. Journal of Neurophysiology. 1992;68(1):100–108.

30. Feinstein B, Lindegård B, Nyman E, Wohlfart G. Morphologic studies of motor units in normal human muscles. Cells Tissues Organs. 1955;23(2):127–142.

31. Caillet AH, Phillips AT, Farina D, Modenese L. Mathematical relationships between spinal motoneuron properties. Elife. 2022;11:e76489.

32. De Luca CJ, Hostage EC. Relationship between firing rate and recruitment threshold of motoneurons in voluntary isometric contractions. Journal of neurophysiology. 2010;104(2):1034–1046.

33. Henneman E. Recruitment of motoneurones: the size principle. Prog Clin Neurophysiol. 1981;9:26–60.

34. Grandi Sgambato B, Hasbani M, Barsakcioglu DY, Ibáñez J, Jakob A, Fournelle M, et al. High Performance Wearable Ultrasound as a Human Machine Interface for wrist and hand kinematic tracking. TechRxiv Preprint. 2023;.

35. Zhao J, Yu Y, Wang X, Ma S, Sheng X, Zhu X. A musculoskeletal model driven by muscle synergy-derived excitations for hand and wrist movements. Journal of Neural Engineering. 2022;19(1):016027.

36. Zhao J, Yu Y, Sheng X, Zhu X. Consistent control information driven musculoskeletal model for multiday myoelectric control. Journal of Neural Engineering. 2023;.

37. Glaser V, Holobar A. Motor unit identification from high-density surface electromyograms in repeated dynamic muscle contractions. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2018;27(1):66–75.

38. Kramberger M, Holobar A. On the prediction of motor unit filter changes in blind source separation of high-density surface electromyograms during dynamic muscle contractions. IEEE Access. 2021;9:103533–103540.

39. Oliveira AS, Negro F. Neural control of matched motor units during muscle shortening and lengthening at increasing velocities. Journal of Applied Physiology. 2021;130(6):1798–1813. doi:10.1152/japplphysiol.00043.2021.

40. Glaser V, Farina D, Holobar A. Simulations of high-density surface electromyograms in dynamic muscle contractions. In: 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). IEEE; 2017. p. 3453–3456.

41. Chen C, Ma S, Sheng X, Farina D, Zhu X. Adaptive real-time identification of motor unit discharges from non-stationary high-density surface electromyographic signals. IEEE Transactions on Biomedical Engineering. 2020;67(12):3501–3509.

42. Wen S, Yin A, Furlanello T, Perich M, Miller L, Itti L. Rapid adaptation of brain–computer interfaces to new neuronal ensembles or participants via generative modelling. Nature Biomedical Engineering. 2021; p. 1–13.

43. Caillet AH, Phillips AT, Carty C, Farina D, Modenese L. Hill-type computational models of muscle-tendon actuators: a systematic review. bioRxiv. 2022; p. 2022–10.

44. Navacchia A, Myers CA, Rullkoetter PJ, Shelburne KB. Prediction of in vivo knee joint loads using a global probabilistic analysis. Journal of Biomechanical Engineering. 2016;138(3):031002.

45. Myers CA, Laz PJ, Shelburne KB, Davidson BS. A probabilistic approach to quantify the impact of uncertainty propagation in musculoskeletal simulations. Annals of Biomedical Engineering. 2015;43:1098–1111.

46. Valente G, Pitto L, Testi D, Seth A, Delp SL, Stagni R, et al. Are subject-specific musculoskeletal models robust to the uncertainties in parameter identification? PLoS One. 2014;9(11):e112625.

47. Millard M, Uchida T, Seth A, Delp SL. Flexing computational muscle: modeling and simulation of musculotendon dynamics. Journal of Biomechanical Engineering. 2013;135(2):021005.

48. Zajac FE. Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control. Critical Reviews in Biomedical Engineering. 1989;17(4):359–411.