Wearable Lower Limb Neuroprosthesis: System Architecture and Control Tuning*

Simão P. Carvalho[1,2] [0000-0001-9666-6832], Joana Figueiredo[1,2] [0000-0001-9547-3051], and Cristina P. Santos[1,2] [0000-0003-0023-7203]

[1] Center for MicroElectroMechanical Systems (CMEMS), University of Minho, Portugal
[2] LABBELS – Associate Laboratory, Braga/Guimarães, Portugal
`id8817@alunos.uminho.pt, joana.figueiredo@dei.uminho.pt, cristina@dei.uminho.pt`

Abstract. The use of functional electrical stimulation (FES) through neuroprosthesis is becoming a promising solution in lower limb neurorehabilitation. However, the wearability constraints and time-consuming tuning of stimulation parameters still limit the daily use of neuroprostheses. This work proposes two major contributions, namely: (i) a conceptual design and technical architecture of a fully wearable lower limb neuroprosthesis; and (ii) a Matlab-OpenSim framework that enables fast subject- and muscle-specific tuning of FES controllers based on OpenSim musculoskeletal models. The validation procedures for this study were divided into three phases: (i) Verification of the system architecture real-time requirements; (ii) evaluation of the reliability of the MATLAB-OpenSim framework for tuning PID controller; and (iii) its subsequent use in the neuroprosthesis control with a healthy subject. The obtained results demonstrated that the neuroprosthesis system was able to meet the real-time requirements, with control and data acquisition call periods below 10 ms. Further findings indicated reliable and stable behavior of the simulation-tuned PID controller with an overshoot of 9.82% and a rise time of 0.063 s. The trajectory tracking control results with the neuroprosthesis corroborated the robustness of the tuned PID controller in tracking the desired ankle trajectory (RMSE =$17.23 \pm 2.97^\circ$ and time delay = 0.21 ± 0.070 s).

Keywords: Closed-loop Control, Functional Electrical Stimulation, Wearable rehabilitation robot

* This work has been supported by the Fundação para a Ciência e Tecnologia (FCT) through the Reference Scholarship under Grant SFRH/BD/147878/2019, the Stimulus of Scientific Employment under Grant 2020.03393.CEECIND, and in part by the FEDER Funds through the Programa Operacional Regional do Norte and national funds from FCT with the SmartOs project under Grant NORTE-01-0145-FEDER-030386. It is also supported under the national support to R&D units grant, through the reference project UIDB/04436/2020 and UIDP/04436/2020.

1 Introduction

Around 250k to 500k people suffer a spinal cord injury (SCI) every year [1]. Sequalae of SCI may include loss of sensory and/or motor control of lower limbs, spasticity, muscle weakness, and pain [2]. Robot-based therapies are a promising complementary possibility to enhance rehabilitation outcomes [3]. Robotic assistive devices such as functional electrical stimulation (FES) stand out as the only intervention that can re-establish muscle function after paralysis and its ability to induce physiological benefits (e.g., muscle strength, counteract muscle atrophy, improved cardiovascular health) [4], [5]. FES acts towards the re-establishment of sensorimotor control by creating alternative neural pathways through the use of devices named neuroprosthesis [6].

During the last decades, several neuroprostheses have been developed targeting different body locations and joint motion (e.g., ankle [7], knee [8], shoulder and elbow [9], knee and ankle [10]). Despite advancing control algorithms, the proposed solutions [7]–[10] are still not compliant for long-term use in daily assistance, given their lack of wearability. The level of motor function recovery of SCI patients obtained from robotics-based rehabilitation is directly dependent on the execution of activities of daily living (ADLs) [4]. In this sense, the development of wearable rehabilitation robots is an untended challenge that is fundamental to reaching task-oriented locomotion assistance adapted to the patient's ADLs and motor progress [11].

Recent studies have proposed promising trajectory tracking control strategies for neuroprosthesis, commonly implemented through PID controllers, to elicit different types of movement [8], [9], [12]. Despite the major advancements suggested in these studies [8], [9], [12], one important challenge remains in the way the controller is tuned to generate the desired stimulation. It requires a subject- and muscle-specific tuning to achieve muscle-specific stimulation profiles, which should be regularly updated given time-variable muscle response. Thus, the process of tuning an FES controller may take a considerable number of experiments, being time-consuming. Further, the tuning is limited by the subject's ability to stand high levels of stimulation for long-lasting experiments [13], being a factor that may cause muscle fatigue.

This work aims to contribute to these two challenges. First, it presents the conceptual design and technological description of a modular and real-time architecture for a fully wearable lower limb neuroprosthesis. The system is formed by a stimulation unit that can electrically stimulate up to 8 muscles, an inertial based-sensor network able to track full lower limb kinematics [14], and a wearable central processing unit (CPU). The technological innovation, when compared to [8], [9], [12], [15], lies in using a wearable CPU and a modular architecture that enables both stand-alone use and direct integration into third-party systems. Overall, the technological description may guide future developments in wearable neuroprosthesis. Additionally, this study provides a Matlab-OpenSim framework that enables fast subject- and muscle-specific tuning of FES controllers, considering the open-source OpenSim musculoskeletal models. These models can be customized to the subject's anthropometric data and include all relevant muscles for lower limb motion. To demonstrate the reliability of the Matlab-OpenSim framework, this study describes the implementation and the tuning of the neuroprosthesis-oriented PID controller for the ankle joint. Experimental procedures, considering the controller parameters obtained from the framework, were conducted with a healthy subject during ankle joint motion evoked by a two-channel neuroprosthesis. The findings demonstrated the potential of the Matlab-OpenSim framework for FES controller tuning.

2 Methods

2.1 Neuroprosthesis Architecture

The neuroprosthesis architecture is composed of a Raspberry Pi 4B (CPU), motionstim8 (stimulation unit), InertialLab [14] (inertial sensor network), and the power supply, as demonstrated in Fig 1.a). Fig 1.b) depicts the fully wearable neuroprosthesis placed on a subject. The power supply, CPU, stimulation unit, and the InertialLab master board are fixed inside the backpack, while the Inertial Measurement Units (IMUs) and electrodes are placed on the lower limbs. The connectivity between the backpack and the electrodes and IMUs is made through cables guided alongside the user's legs. Additionally, an Android App, running on a smartphone, was developed to configure, start, and stop the system. A description of each module is presented below.

2.1.1 Central Processing Unit

The CPU is a Raspberry Pi 4 Model B (Raspberry Pi Foundation, UK), a single-board computer with a Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz and 4 GB of RAM. It has 56.5x85.6x11 mm and weighs 46 g. The InertialLAB and the stimulation unit communicate with the CPU at 100 Hz through Universal Asynchronous Receiver/Transmitter (UART) using a USB converter (FT232RL FTDI). The CPU uses the Ubuntu Mate OS to run the control strategy at 100 Hz, using the C++ programming language.

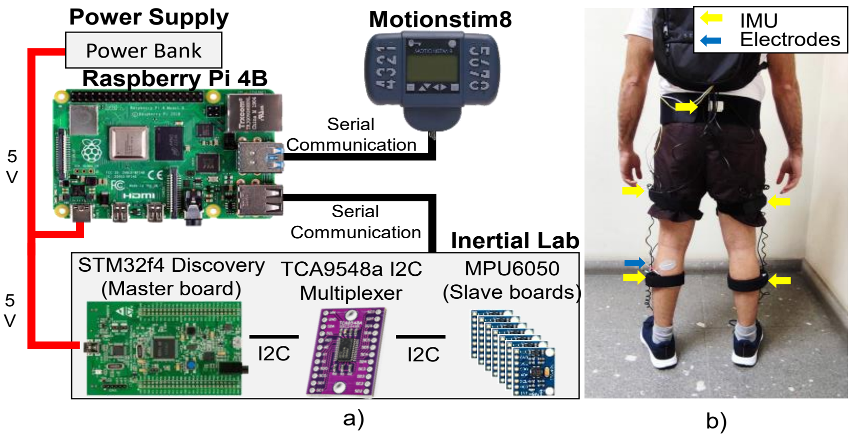


Fig 1. a) Schematic overview of the neuroprosthesis system. b) Subject wearing the system with 7 IMUs and 2 stimulation channels.

The software architecture of the CPU (depicted in Fig. 2) was organized into four main software modules (classes), namely: (i) *CentralController* (in charge of setup, start, and stop all configurable modules), (ii) *ExternalDevice* (manages the communication between the CPU and external devices), (iii) *Monitor* (handles the data log), and (iv) *NPController* (manages the assistive control strategy). Each class has a thread based on POSIX Pthread Library. The *CentralController* is the main class and, therefore, inherits all the other classes. Through an Android APP, which communicates with the *CentralController* via Bluetooth, the user can easily configure the system (i.e., number and location of sensors, number and location of channels, gait speed, type of control, control settings, user's anthropometric data, among others), and start and stop the therapy. Given the system modularity, the user can select any sensor and channel as well as limit the therapy to data monitoring (i.e., without stimulation). Further, the APP can also provide feedback to the user in case of unexpected system errors. The *ExternalDevice* class receives the user commands and configures the external devices accordantly (i.e., the inertial sensor network and stimulation unit). During the therapy, the *ExternalDevice* handles the communication with these two devices through tasks of Pthread set at 100 Hz. The *Monitor* class guarantees that all sensor and control-specific variables are saved, every 10 ms, in the CPU memory in text files for posterior data analysis.

The *NPController* class has three hierarchically organized subclasses, namely: HLController, MLController, and LLController. This control architecture is inspired by the principles and organization of the human motion-control system [16]. The high-level, perception layer was designed to decode the user's locomotion intentions, locomotion status, and disability level. The mid-level, the translation layer, generates and adapts position reference trajectories to the user's needs and intentions. Lastly, the low-level layer runs PID-based position tracking controllers to generate assistive commands to the stimulation unit. This architecture presents a modular design to be expandable for including further assistive control strategies. A timer sets up the *NPController* tasks execution at 100 Hz.

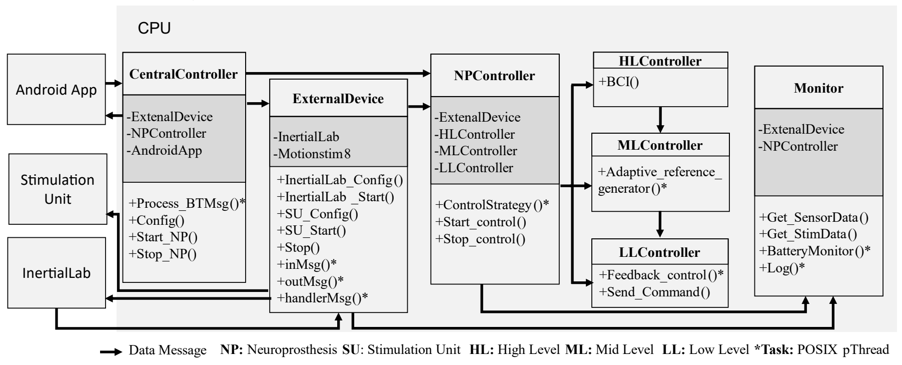


Fig. 2. Neuroprosthesis software architecture diagram - classes and their interconnectivity.

2.1.2 Power Supply

The power supply system includes two standard power banks (10000 mAh, 5V) to power the InertialLab and CPU and allows at least 4 hours of autonomy. Together, the body mass (360 g) and dimensions (92x60x44 mm) of the power banks match the requirements for system wearability. The stimulation unit has its own-integrated battery.

2.1.3 Inertial Sensor Network

The inertial sensor network is the InertialLab [14], which allows to set up from 1 up to 7 IMUs for real-time acquisition at 100 Hz. It includes the STM32F4-Discovery development board (STMicroelectronics, Switzerland) that communicates with the CPU through a UART interface. Each IMU consists of the MPU-6050 (InvenSense, Boston, MA, USA) that combines a 3-axis MEMS accelerometer (±8 g) and a 3-axis MEMS gyroscope (±2000/s) for the kinematic data acquisition. The MPU-6050 sensor communicates with the STM32F4-Discovery via $\mathrm{I^2C}$ through the TCA9548A multiplexer to manage the multi-channel data collection [14].

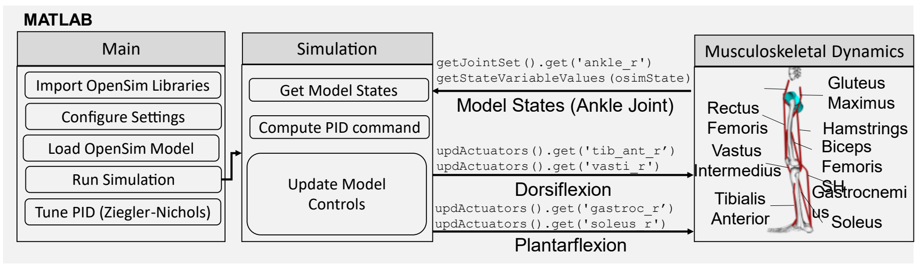
The InertialLab provides the raw data for each IMU (i.e., XYZ acceleration and XYZ angular velocity), up to seven segment angles (i.e., trunk, right/left thigh, right/left shank and right/left foot), and up to six joint angles (i.e., right/left hip, right/left knee, and right/left ankle). The segment and joint angle estimation are described in [14].

2.1.4 Stimulation Unit

The Motionstim8 (MEDEL GmbH, Medicine Electronics, Hamburg, Germany) is an electrical stimulation device that allows the individual control of up to 8 stimulation channels simultaneously. This device may be manually controlled or may receive commands from a third-party device (i.e., science mode). For the present work, the Motionstim8 was used in the science mode. It communicates with the CPU using a serial communication protocol (baud rate = 115200). The electrical stimulation pulses are rectangular and biphasic and can be controlled regarding their intensity (1 - 125 mA), pulse width (10 - 500 μsec), and frequency (1 - 99 Hz). The stimulation intensity and frequency were found empirically, considering the pain threshold of the subject and the range of motion of the ankle, as in [17]. The configuration of the stimulation unit was set up to 30 Hz of pulse frequency, 30 mA of pulse intensity, and the pulse width as the control command. The pulse width is computed in the low-level control using the trajectory tracking control strategy, as depicted in Fig. 4.

2.2 Matlab-OpenSim Framework

A simulation framework was implemented in MATLAB R2021a, considering its integration with the OpenSim environment. The OpenSim platform allows the execution of the forward dynamics by providing the muscle excitations that generate the musculoskeletal model's motion [18]. The developed framework is represented in Fig. 3. It is formed by the *Main* script, a recursive *Simulation* function, and the musculoskeletal model.



```
getJointSet().get('ankle_r')
getStateVariableValues(osimState)
updActuators().get('tib_ant_r')
updActuators().get('vasti_r')
updActuators().get('gastroc_r')
updActuators().get('soleus_r')
```

Fig. 3. Schematic overview of the Matlab-OpenSim framework.

The *Main* script starts by including the OpenSim libraries (*org.opensim.modeling)*. Then, it loads the *Gait10dof18musc* OpenSim model and allows the user to configure the stimulation frequency, activate/deactivate the PID tuning, change the reference trajectory signal, and set up PID gains. To conduct the simulations, the *Gait10dof18musc* OpenSim model was pinned to prevent it from being pulled by gravity, as if it was hanging by a harness. In addition, all joints were locked except for the ankles to prevent residual movements. The model was selected considering its computation efficiency, given its simplified muscle anatomy, which includes the main nine muscle groups in each leg, namely: *Hamstrings*, *Biceps Femoris Short Head*, *Gluteus*, *Iliopsoas*, *Rectus Femoris*, *Vastus Lateralis*, *Gastrocnemius*, *Soleus* and *Tibialis Anterior*. At this stage, the forward dynamics simulation is configured. The simulation function is defined with the *controlFunctionHandle* and the integration function is the *ode15s*. At the end of the simulation, the PID tunning function executes (when activated), automatically providing the PID gains.

The *Simulation* function executes recursively at 100 Hz, and each iteration considers the model states (i.e., joint angles) and the reference trajectory to compute the position error. The position error is used to compute the PID command, which is saturated between -1 and 1. The function *Update Model Control* sends the excitation signals (i.e., PID commands) to the dorsiflexion and plantarflexion muscles of the *Gait10dof18musc* OpenSim model.

The tunning process was conducted for the ankle joint, targeting the *Vastus Lateralis*, *Tibialis Anterior* for ankle dorsiflexion, and *Gastrocnemius*, *Soleus* for ankle plantarflexion. For this purpose, a step trajectory of 50º was used. The integral and derivative gains were set to zero and the ultimate (proportional) gain was obtained after achieving a stable oscillation of the ankle around the reference position ($\mathrm{Kp} = 5, \mathrm{Ki} = 0, \mathrm{Kd} = 0$). Subsequently, the PID gains were computed using the Ziegler Nichols method.

2.3 Neuroprosthesis Control

The neuroprosthesis control architecture is hierarchically structured into three layers, namely: (i) High-level; (ii) Mid-level; and (iii) Low-level, as mentioned in section 2.1.1. The current neuroprosthesis development stage only includes the mid and low-level layers; however, the architecture is prepared for the addition of the high-level in further developments. The high-level layer will be composed of a brain-computer interface capable of decoding the user's lower limbs motion intention, which will subsequently and automatically trigger the electrical stimulation. The mid-level includes the regression model proposed in [19] that generates user-oriented position trajectories for the hip, knee, and ankle joints according to the user's height and gait speed (both indicated through the Android App). This regression model scales the amplitude of the trajectory attending to the user's height, and the amplitude and timing of the trajectory based on the gait speed [19]. The low-lever controller covers a trajectory tracking control strategy relying on a feedback controller (PID). The PID includes an anti-windup strategy to reset the integral component when a zero-crossing occurs between the measured angle and the reference angle. In addition, when the controller hits its saturation limits, an anti-windup back-calculation method is used to discharge the integrator component. In this context, the plant is the human muscle, and to control the dorsiflexion (upward movement of the foot) and plantarflexion (downward movement of the foot), two stimulation channels are required in the *Tibialis Anterior,* and *Gastrocnemius*, respectively. In this sense, the PID is built to send commands to the dorsiflexion channel if $e_k > 0$, and to send commands to the plantarflexion channel if $e_k < 0$, as demonstrated in Fig. 4. The PID was tuned using the MATLAB-OpenSim framework.

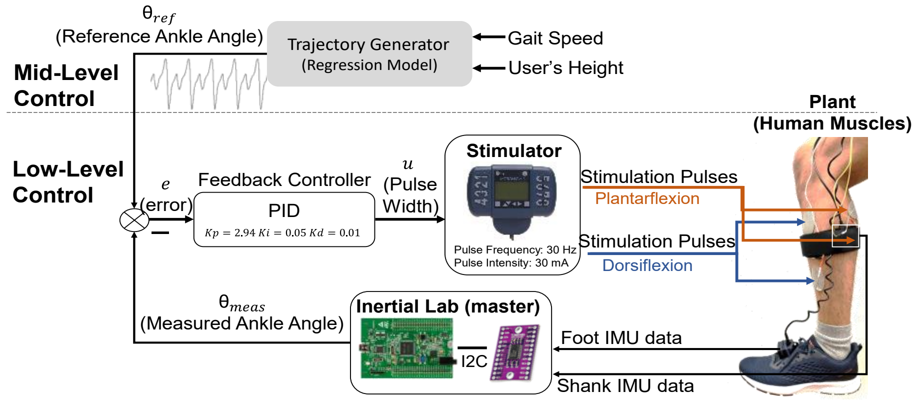


Fig. 4. Neuroprosthesis' control architecture, depicting the low- and mid-level layers.

2.4 Validation

This study has issued three validation phases. First, we conducted bench tests to verify whether the implemented architecture is able to fulfill the real-time requirements for inertial data acquisition and control every 10 ms. The validation was performed for 10 minutes when acquiring data from the 7 IMUs and simulating the 8 channels. During the validation, the timing of the call periods of each class was collected with a digital oscilloscope for a posterior analysis (mean and standard deviation).

The second and third phases aimed at evaluating the reliability of MATLAB-OpenSim framework for tuning the PID controller, and its subsequent use in the neuroprosthesis control in a real environment, respectively. In the second phase, the response of the PID controller obtained in the simulation was tested with a step reference of -50º and a sinusoidal reference trajectory (frequency of 1Hz, a peak amplitude of 20º, and an offset of -20º). The step response was analyzed with overshoot percentage, rise time, and settling time, considering the maximum requirements (i.e., overshoot<10%, rise time<0.2 s, settling time<2 s). On the other hand, the response of the controller to a variable trajectory (sinusoidal) was analyzed with the RMSE, normalized RMSE, and time delay, to determine its reliability to be used in the neuroprosthesis.

In the third phase, we tested if the PID gains found in the MATLAB-OpenSim framework enable a reliable stimulation profile for controlling the human ankle motion. This validation analysis was based on the RMSE, time delay, and their normalized values. This phase involved a healthy male subject 27 years old, 70 kg, and 182 cm in height. The subject gave his informed consent according to the ethical conduct defined by the University of Minho Ethics Committee (CEICVS 006/2020) following the standards set by the declaration of Helsinki and the Oviedo Convention.

The participant was instrumented as follows. For the dorsiflexion channel, one oval electrode (4x6 cm) was positioned just below the fibular head, above the *tibialis anterior* muscle belly. The second oval electrode (4x6 cm) was placed about 2/3 of the way down the shin with the leads facing toward the midline of the body. Regarding the plantarflexion channel, one oval electrode (4x6 cm) was placed just below the knee, and the second electrode (4x6 cm) two fingers below the first electrode, in the belly of the *soleus* muscle (see Fig. 4). Further, he wore an IMU on the center of the left foot and another IMU on the lateral side of the left shank (see Fig. 4) by using straps.

Before initializing the experiment, a procedure was conducted to identify the maximum painless stimulation pulse width values. The limits for the dorsiflexion and plantarflexion channels were set to 250 μs and 100 μs, respectively. Afterwards, the PID testing experiment with the neuroprosthesis was conducted with the subject in the upright position while having his weight partially supported by a harness system. The experiment included 8 trials, in which the neuroprosthesis control was actively stimulating the subject for 5 minutes. Between each trial, the subject rested for 2 minutes.

3 Results and Discussion

3.1 Phase I – System Architecture

Table 1 presents the results of the computational performance of the neuroprosthesis architecture, for the inertial data acquisition and control. This evaluation considered the timers and tasks evocated repeatedly in real-time to investigate the effects of a non-centralized architecture during real-time therapy.

Table 1. Computational performance evaluation

Task	Requirement (ms)	Measures	
		Mean ± STD (ms)	RMSE (ms)
Control	10	9.999 ± 0.002	0.003
Data Acquisition	10	9.998 ± 0.005	0.007

Table 1 reports RMSE values lower than 0.003 ms and 0.007 ms regarding the control and data acquisition tasks, respectively. These results indicate that the time requirements (10 ms) for the system architecture execution were met. Further, there is variability in the call period times along the therapy execution, as shown by STD values presented in Table 1. This variability was more evident during the system initialization and is higher for the data acquisition task, given the higher processing complexity for the InertialLab.

3.2 Phase II - Matlab-OpenSim Framework

The tunning process using the MATLAB-Opensim framework revealed that the ankle joint of the musculoskeletal model started to present a constant oscillatory response when the proportional gain (K_p) reached the value 5 (ultimate gain). Fig. 5.(a) demonstrates the oscillatory angle response to a step reference of -50º.

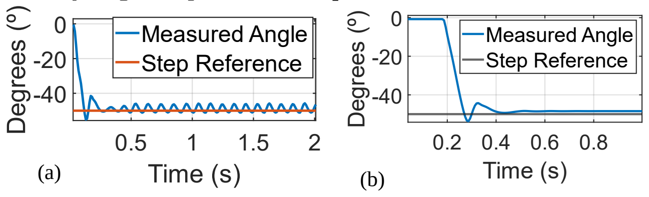


Fig. 5. a) Simulated oscillatory response of the ankle under a proportional controller ($K_p = 5$). b) Simulated step response of the ankle using the tunned PID (Kp=2.94, Ki=0.05, Kd=0.01)

From the Ziegler Nichols classic tunning rules, the final PID gains were determined and corresponded to Kp=2.94, Ki=0.05, Kd=0.01. The PID controller was subsequently tested with these gains, considering a step response of -50º. The resulting ankle joint response of the musculoskeletal model is demonstrated in Fig. 5. (b).

The step response analysis reported an overshoot percentage of 9.82%, a rise time of 0.063 s, and a settling time of 1.7042 s, indicating that the system was able to meet the specified requirements (overshoot<10%, rise time<0.2s, and settling time<2s). Further, the reduced oscillatory response indicates the high stability of the system. The obtained step response suggests that the controller is suited to be used in the real setting, considering the improved results in overshoot (<8%) and rise time (<1s), regarding a similar PID tuning approach for a different human joint [13].

Further validation procedures were conducted in simulation for the PID controller, considering a sinusoidal reference trajectory.

Fig. **6** presents the results obtained including the trajectory tracking response, the tracking error, and the PID command sent to the OpenSim musculoskeletal model.Results reported a RMSE of 0.029º, a Normalized RMSE of 3.905%, and a time delay of 0.01s. These results suggest a robust tracking performance using a variable reference trajectory, demonstrating reduced muscle excitation values associated with reduced RMSE and time delay, which stands as an ideal scenario.

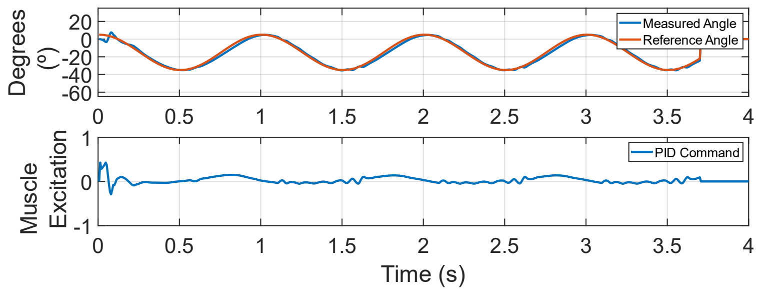


Fig. 6. Simulated ankle behaviour for the sinusoidal reference using the tuned PID.

3.3 Phase III - Neuroprosthesis Control

The neuroprosthesis trajectory tracking control was subsequently validated using the PID gains determined with the MATLAB-Opensim framework. Fig. 7.a) represents the reference trajectory (blue line) and the measured ankle joint angle (red line) considering the execution of three gait cycles. For this experiment, a healthy ankle angle trajectory was used as the reference. Fig. 7.b) shows the generated pulse width commands for the dorsiflexion channel (blue line) and the plantarflexion channel (red line).

From the neuroprosthesis control validation, performance results were obtained, namely, a RMSE of 17.23±2.97º, a normalized RMSE of 3.97±0.69%, a time delay of 0.21±0.070 s, and a normalized time delay of 0.0044±0.0014 s. The pulse width values tended to increase with time as a compensation for the increased muscle fatigue. Similar behaviors were reported in previous studies using FES [12]. These preliminary results suggest a promising reference tracking, presenting normalized RMSE values bellow literature (3.97% <16.7%) reported values under a similar control approach for the knee joint [13]. Additionally, the neuroprosthesis controller was able to maintain the tracking delay below literature values (250 ms) [12], which stands as a challenge in FES controllers due to the existing natural electromechanical muscle delay (10-100 ms) [20].

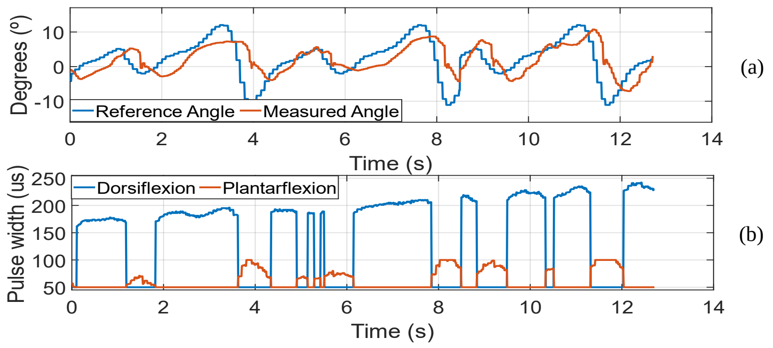


Fig. 7. Example of three gait cycles of Neuroprosthesis tracking performance with the healthy subject using the PID tuned by the MATLAB-OpenSim framework.

4 Conclusions

In this paper, we presented the modular architecture of a neuroprosthesis that tackles the lack of wearable neuroprosthetic solutions. The tests indicated that the neuroprosthesis architecture successfully meets the real-time requirements set for gait analysis and rehabilitation. Further, we proposed a MATLAB-OpenSim-based framework to enable a fast subject- and muscle-specific tuning of FES controllers. To demonstrate the reliability of the framework, this study evaluated the tuned PID controller in a real neuroprosthesis when controlling the ankle joint. Results reported reduced RMSE and time delay of the tuned PID controller in the MATLAB-OpenSim-based framework. Overall, this shows the potential of the MATLAB-OpenSim-based framework to tune further controllers of wearable neuroprostheses. Future developments include scaling the neuroprosthesis to the remaining lower limb joint as well as extending the validation to more subjects.

References

[1] W. H. Organization, "WHO | International perspectives on spinal cord injury," *WHO*, 2013, Accessed: Apr. 22, 2020. [Online]. Available: https://www.who.int/disabilities/policies/spinal_cord_injury/en/.

[2] C. Marquez-Chin, I. Bolivar-Tellería, and M. R. Popovic, *Brain–computer interfaces for neurorehabilitation: enhancing functional electrical stimulation*, Second Edi. Elsevier B.V.

[3] J. C. Moreno, J. Figueiredo, and J. L. Pons, "Exoskeletons for lower-limb rehabilitation," in *Rehabilitation Robotics: Technology and Application*, London, United Kingdom: Elsevier, 2018, pp. 89–100.

[4] J. C. Moreno, S. Mohammed, N. Sharma, and A. J. Del-Ama, *Hybrid wearable robotic exoskeletons for human walking*. INC, 2019.

[5] V. Molazadeh, Q. Zhang, X. Bao, B. E. Dicianno, and N. Sharma, "Shared Control of a Powered Exoskeleton and Functional Electrical Stimulation Using Iterative Learning," *Frontiers in Robotics and AI*, vol. 8, no. November, pp. 1–13, 2021, doi: 10.3389/frobt.2021.711388.

[6] M. Shah, C. Peterson, E. Yilmaz, D. R. Halalmeh, and M. Moisi, "Current advancements in the management of spinal cord injury: A comprehensive review of literature," *Surgical Neurology International*, vol. 10, no. 174, pp. 1–4, 2019, doi: 10.25259/SNI.

[7] T. Seel, D. Laidig, M. Valtin, C. Werner, J. Raisch, and T. Schauer, "Feedback control of foot eversion in the adaptive peroneal stimulator," in *22nd Mediterranean Conference on Control and Automation*, Jun. 2014, pp. 1482–1487, doi: 10.1109/MED.2014.6961585.

[8] M. Ferrarin, F. Palazzo, R. Riener, and J. Quintern, "Model-based control of FES-induced single joint movements," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 9, no. 3, pp. 245–257, 2001, doi: 10.1109/7333.948452.

[9] F. Resquín, J. L. Pons, F. Brunetti, J. Ibáñez, and J. Gonzalez-Vargas, "Feedback error

learning controller for functional electrical stimulation assistance in a hybrid robotic system for reaching rehabilitation," *European Journal of Translational Myology*, vol. 26, no. 3, pp. 255–261, 2016, doi: 10.4081/ejtm.2016.6164.

[10] P. Müller, A. J. Del Ama, J. C. Moreno, and T. Schauer, "Adaptive multichannel FES neuroprosthesis with learning control and automatic gait assessment," *Journal of NeuroEngineering and Rehabilitation*, vol. 17, no. 1, pp. 1–20, 2020, doi: 10.1186/s12984-020-0640-7.

[11] H. Kim, G. Park, J. H. Shin, and J. H. You, "Neuroplastic effects of end-effector robotic gait training for hemiparetic stroke: a randomised controlled trial," *Scientific Reports*, vol. 10, no. 1, pp. 1–9, 2020, doi: 10.1038/s41598-020-69367-3.

[12] A. Pedrocchi, S. Ferrante, E. De Momi, and G. Ferrigno, "Error mapping controller: A closed loop neuroprosthesis controlled by artificial neural networks," *Journal of NeuroEngineering and Rehabilitation*, 2006, doi: 10.1186/1743-0003-3-25.

[13] A. C. C. de Sousa, F. M. Ramos, M. C. Narvaez Dorado, L. O. da Fonseca, and A. P. Lanari Bó, "A Comparative Study on Control Strategies for FES Cycling Using a Detailed Musculoskeletal Model," *IFAC-PapersOnLine*, vol. 49, no. 32, pp. 204–209, 2016, doi: https://doi.org/10.1016/j.ifacol.2016.12.215.

[14] J. Figueiredo, S. P. Carvalho, J. P. Vilas-Boas, L. M. Gonçalves, J. C. Moreno, and C. P. Santos, "Wearable inertial sensor system towards daily human kinematic gait analysis: Benchmarking analysis to MVN BIOMECH," *Sensors (Switzerland)*, vol. 20, no. 8, p. 2185, Apr. 2020, doi: 10.3390/s20082185.

[15] P. Müller *et al.*, "Adaptive multichannel FES neuroprosthesis with learning control and automatic gait assessment," *Journal of NeuroEngineering and Rehabilitation*, vol. 17, no. 1, pp. 1–20, Feb. 2020, doi: 10.1186/s12984-020-0640-7.

[16] M. R. Tucker *et al.*, "Control strategies for active lower extremity prosthetics and orthotics: A review," *Journal of NeuroEngineering and Rehabilitation*, vol. 12, no. 1, 2015, doi: 10.1186/1743-0003-12-1.

[17] M. Bouri, A. Selfslagh, D. Campos, S. Yonamine, A. R. C. Donati, and S. Shokur, "Closed-Loop Functional Electrical Stimulation for Gait Training for Patients with Paraplegia," *2018 IEEE International Conference on Robotics and Biomimetics, ROBIO 2018*, pp. 1489–1495, 2018, doi: 10.1109/ROBIO.2018.8665270.

[18] A. Seth *et al.*, "OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement," *PLOS Computational Biology*, vol. 14, no. 7, p. e1006223, Jul. 2018, doi: 10.1371/JOURNAL.PCBI.1006223.

[19] B. Koopman, E. H. F. van Asseldonk, and H. Van der Kooij, "Speed-dependent reference joint trajectory generation for robotic gait support," *Journal of Biomechanics*, vol. 47, no. 6, pp. 1447–1458, 2014, doi: 10.1016/j.jbiomech.2014.01.037.

[20] P. R. Cavanagh and P. V. Komi, "Electromechanical delay in human skeletal muscle under concentric and eccentric contractions," *European journal of applied physiology and occupational physiology*, Nov. , doi: 10.1007/BF00431022.