IDENTIFICATION OF MOTION-BASED ACTION POTENTIALS IN NEURAL BUNDLES USING A CONTINUOUS SYMBIOTIC SYSTEM

Volkhard Klinger
Department of Embedded Systems
University of Applied Science Hannover (FHDW)
30173 Hannover, LS, Germany
volkhard.klinger@fhdw.de

Sebastian Bohlmann Helena Szczerbicka
Department of Simulation and Modeling
Leibniz University Hannover
30167 Hannover, LS, Germany
bohlmann@ifam.uni-hannover.de
hsz@sim.uni-hannover.de

ABSTRACT

The identification of motion- and sensory feedback-based action potentials in peripheral nerves is a great challenge in medical technology. It is the prerequisite for applications like prosthesis control or limb stimulation. Based on the acquisition of action potentials, the identification process correlates physiological and motion-based parameters to match movement trajectories and the corresponding action potentials. In this paper we focus on the identification method based on a data driven approach and its verification. We present the closed-loop identification method, implemented using a symbiotic continuous system (Aydt, Turner, Cai, and Low 2008), (Aydt, Turner, Cai, and Low 2009), consisting of a robotic based trajectory generation, the nerve simulation and, an agent-based machine learning system. We introduce the model generation process, showing an emergent behavior and present results of different scenarios generated using synthetic data sets. We show the whole verification approach of the identification method and illustrate the influence of the identification parameters on the quality of results.

Keywords: symbiotic simulation, symbiotic loop, system identification, agent-based evolutionary computation.

1 INTRODUCTION

Today prosthesis is even more than only easy spare parts for the human body. From the simple wooden butts of the past ingenious high-tech constructions have become. However, the modern medicine can substitute even more than only arms and legs. The main problem is the human machine interface of prosthesis and its movement control. The objective is to use biosignals for the information transfer between human being and prosthesis. So an interface is needed to interfere between the command-level and the actuator- and sensor- level. Several possibilities are existing to realize such an interface and to link to the command-level: The electroencephalogram (EEG), the electromyogram (EMG) and the electroneurogram (ENG). The use of EEG-signals is not applicable in the most cases due to the fact, that the EEG-signal is buried within other brain activity and therefore quite indifferent according to the specific movement commands. Another option is the sampling of the EMG activity of several selected muscles. One disadvantages is that the signal activity responsible for the movement of the forearm and the hand is distributed over lots of muscles. Another drawback is the absence of the corresponding muscle groups in the case of an amputation.
The approach discussed in this paper is based on the direct use of the action potentials of peripheral neural bundles via an ENG (Gold et al., 2007; Neymotin et al., 2011). Based on these signals, a prosthesis, for

example, an artificial hand or an artificial forearm, can be controlled specifically. An interesting side effect when the ENG is used directly is a high probability that the nerve functionality can be preserved from degeneration in the case of an amputation.
One central aspect of this approach is the minimal-invasive character, simplifying the implantation and it provides more flexibility with regard to inter- and intra-individual differences. So, the employment of invasive intra-neural sensors (Micera, Citi, Rigosa, Carpaneto, Raspopovic, Pino, Rossini, Yoshida, Denaro, Dario, and Rossini 2010), (Micera, Carpaneto, and Raspopovic 2010), is in this project not in the focus, but the identification (Cesqui, Tropea, Micera, and Krebs 2013) of motion-based action potentials is the proposal to realize a smart minimal-invasive solution. To record the very small ENG-signals, which are only of the order of a few microvolts, we have designed a special front-end hardware/software system realized in two different prototypes, introduced in (Klinger and Klauke 2013) and (Klinger 2015).
In this paper we focus on a new verification method, taking advantage of a continuous symbiotic system (Aydt, Turner, Cai, and Low 2008), (Aydt, Turner, Cai, and Low 2009). At first we present the background application and the used measuring electrode. In the following section we introduce the whole system concept and some applications. Then we compare different verification methods and show several results. This work continues the former work about system identification presented in (Bohlmann, Klauke, Klinger, and Szczerbicka 2011), (Bohlmann, Klinger, and Szczerbicka 2009) and (Bohlmann, Klinger, and Szczerbicka 2010).

2 SYSTEM ARCHITECTURE

At first we introduce the used cuff electrode and present subsequently the system architecture.

2.1 The Cuff-Electrode

This type of electrode encloses the neural bundle to be examined. The single electrodes, part of the cuff-electrode, are inserted in a matrix formed with biomedical silicone. This silicone protects the single electrodes and fix them within the right position. The main advantage of such an electrode is the non-invasive character. Properly designed this electrode can be used without cutting any nerves within a neural bundle. In contrast using a sieve electrode the probability of an irreparably damage of the nerves is very high.
The cuff-electrode used in this application is a special cuff-electrode depicted in Figure 1. It consists of several electrodes organized in rings and segments. In the minimal configuration there are three segments (120°) and three rings necessary. The three electrodes depicted there are evenly distributed on the circumference of the cuff (0°, 120°, 240°) to guarantee an uniform coverage of the action potentials triggered by the neural bundle for example via triangulation. Higher order of rings and segments depends on the space for the implanted cuff-electrode and the best available precision for electrode manufacturing. The number of axons of a neural bundle ranges up to several tens of thousands depending on the type of neural bundle and the selected application localization, like nervus ischiadicus, nervus radialis or nervus medianus.

2.2 System Architecture

The Smart Modular Biosignal Acqusition, Identification and Control System (SMoBAICS), shown in Figure 2, integrates all necessary tasks (Hazan, Zugaro, and Buzsáki 2006). The biosignal acquisition is done by the Modular Biosignal Acquisition System (MBASY)-subsystem, the next generation of our own frontend-hardware/software-system; the first measurement results are described in (Bohlmann, Klauke, and Klinger 2013). The MBASYis redesigned to get a better functionality and to optimize the modular concept. The central part of the identification process is integrated in the Biosignal Identification and Control System) (BICS). It consists of two parts: the machine learning & identification and the prostheses control/stimulation. While

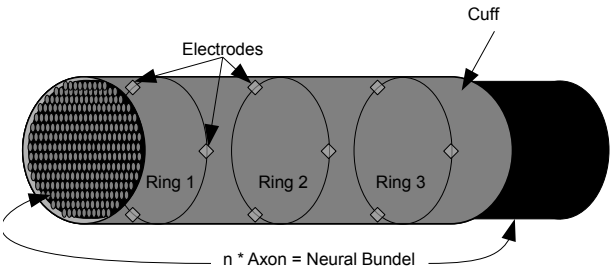

Figure 1: The Cuff Electrode.

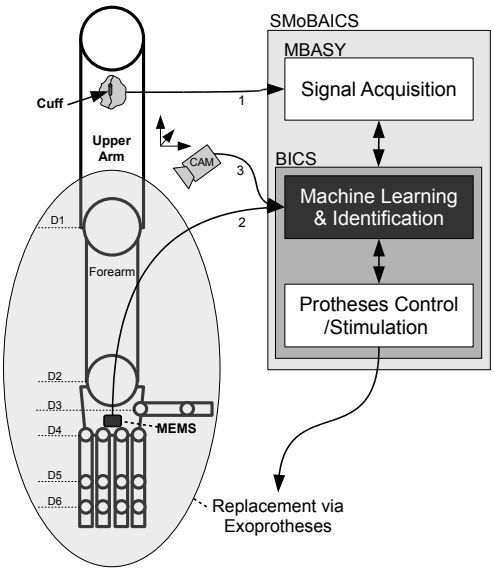

Figure 2: System overview.

the second part is designed by state-of-the-art technology, the machine learning & identification is composed of multi-agent-based optimization algorithm and an evolutionary correlation of different types of nerve signals and of additional information like camera positioning or micro-electro-mechanical systems (MEMS). We introduce this part in more detail in section 3.

Before describing the overall function we have to take one aspect regarding the learning procedure into consideration. The objective of SMoBAICS is the action potential based control or stimulation of upper or lower limbs of handicapped human beings. SMoBAICS provides not only a base identification step (learning phase) but an ongoing supervision (operating phase). Obviously, the operating phase has to be executed on a small body mounted system but in this paper we do not focus on this system detail.

On the left side a typical application is shown, the cuff electrode is implanted in the upper arm enclosing a neural bundle. The action potentials are recorded by the MBASY and passed to the BICS (1). Two additional information streams are used by the BICS for the action potential based identification: a camera-based motion capturing, used during the learning phase (3), and a motion tracking device at the end effector (e.g. the hand), used during the operational phase (2). These data streams are important for the identification.

3 IDENTIFICATION & MACHINE LEARNING

The machine learning and identification is the most complex module within SMoBAICS. There are two different approaches used to check the performance of different methods. In this paper we focus on different verification approaches, providing a new continuous symbiotic method including machine learning. In the following sections we introduce both methods; first we introduce the data preprocessing, part of both methods.

3.1 Data Preprocessing

All real data, recorded from the cuff electrode have to be preprocessed to improve the data conditioning. The verification data generated from the NEURON-simulator can be used directly.

- Filtering: The recorded action potentials are disturbed by intrinsic noise. In addition these are overlaid by a substantial extrinsic noise, originated for example by EMG from surrounding muscles.

Therefore we have to filter the recorded data with integrated analogue filter and additional digital filter.

- Re-sampling: The recorded data has two main weaknesses: The samples are asynchronous and aperiodic. In order to get a time series of data samples we perform the following steps:
  - Interpolation and FIR Filter (finite impulse response)
    For each sequence we interpolate the given values and smooth the result with a convolution.
  - Error Correction
    The interpolated data is equalized with the original samples gained from the Data Factory.
  - Down-sampling
    We pick Euclidean equidistant samples from each sequence and combine them to data samples with a time-stamp.

  During the machine process the data samples will not stay in their chronological ordering. To be able to perform time derivation, it is necessary to save the chronological neighbors for each sample. The resulting time series of equidistant data samples p consists of a time-stamp p_{time}, a vector $p_{\text{data}} = [p_{\text{out}}, p_{\text{in}}]$, with $p_{\text{out}} \in \mathbb{R}$ and $p_{\text{in}} \in \mathbb{R}^m$, containing the output and input data and its chronological neighbors p^{pre} and p^{post}. With P we denote the set of all such data samples. Furthermore we define the delta value $p^{\Delta}_{\text{in}} \in \mathbb{R}$ with

$$(p^{\Delta}_{\text{in}}) := \frac{1}{2}\left(\frac{(p_{\text{in}}) - (p^{\text{pre}}_{\text{in}})}{p_{\text{time}} - p^{\text{pre}}_{\text{time}}} + \frac{(p_{\text{in}}) - (p^{\text{post}}_{\text{in}})}{p_{\text{time}} - p^{\text{post}}_{\text{time}}}\right), \qquad p^{\Delta}_{\text{out}} := \frac{1}{2}\left(\frac{p_{\text{in}} - p^{\text{pre}}_{\text{in}}}{p_{\text{time}} - p^{\text{pre}}_{\text{time}}} + \frac{p_{\text{in}} - p^{\text{post}}_{\text{in}}}{p_{\text{time}} - p^{\text{post}}_{\text{time}}}\right).$$

3.2 Identification via Agent-based Curve Fitting

We use an agent based algorithm to reconstruct the nerve signals, measured with the cuff-electrode. Moreover we can assign certain clusters in the neural bundle to muscle groups and the corresponding receptors. The algorithm is subdivided into three levels, which are described below.

- First-Level
  At the entry level software agents are used to build partial solutions. Every agent has a relatively small scope of data, the size of this data scope is automatically adapted to the extent of parallelization. The task of each agent is the approximation of the data in its scope using its own parameter set containing an action potential template, which takes the physiological restrictions into account, and a model of the neural bundle as well as one of the cuff-electrode. Essentially three decisions are necessary to build a partial solution. For a given piece of input data the agent has to decide which axons where activated (spatial allocation), when it was activated (temporal allocation) and whether it was an actuatory or a reactuatory signal (directional allocation).
- Second-Level
  In this algorithm level the partial solutions found by the software agents in the first level are combined to global solutions. The combination to global solutions is done parallelized by Global Solution Builders (GSB). The agents in the first level are permanently sending new partial solutions for different sections of the input data. Once the whole data set is covered by partial solutions a GSB starts to combine them to a global solution, while the remaining GSBs continue to receive partial solutions. The main task of a GSB is to combine these parameter sets, such that the overall error stays as small as possible.
- Third-Level
  In this level the causal verification of the global solutions takes place. This is done by the Global Solution Tester (GST). So far the solutions were only evaluated according to the pure approximation error. But there are more criterion already to achieve good solutions. The first one is the causal

relationship between actuatory or reactuatory signals. The second criterion is the error with respect to the data from the camera and the MEMS using inverse kinematics. The action potentials from the neural bundle including motor and sensory axons have to be correlated with the results of the inverse kinematics.

3.3 Identification via Machine Learning based on Continuous Symbiotic System

The machine learning is based on evolutionary algorithms – a generic population-based metaheuristic optimization algorithm inspired by biological evolution (De Jong 2006) – embedded in a multi-stage and multi-agent implementation, shown in Figure 3. The planet structure models the environment for the populations inside the evolutionary algorithm. Every planet provides a data field, the software agents can operate on. The number of planets is scalable, the current predetermined size is $n = 9^4 = 6561$. Using a multiprocessor system, the number of planets have to be multiplied by the number of cores. Data acquired from the process connection are preprocessed, equal to the filling of the planet structure. The preprocessing consists of several steps to guarantee a high average information content of the data, so called data entropy. One step is an appropriate data preprocessing, described in principle in subsection 3.1. One further step is the data filling. This last step of the data preprocessing arranges the data samples on a 2D surface of a so-called planet. The surface of the planets is built in a recursive pattern of squares containing nine elements, filled meander like. This method leads to the planet size 9^4. This arrangement has the advantage, that the data set used for the local optimization consists of data samples, which may be spread more widely across the input sequences.

In the central part of figure 3 the agent factory, the model library and the 2-step optimization, dedicated to every planet, is shown. The agents have 4 essential features: an age, an energy level, an area and their model function, approximating the corresponding process function. Moreover a replication mechanism is implemented, meaning the agents are able to produce a child and put it on an area. The age and the energy level are increased after each iteration. All operations an agent can perform, have an energy effort, by which the energy level is lowered, if the operation is executed. Furthermore the agents have the ability to learn from their local data and improve their model function by executing different evolutionary operations to change the structure of the model function and a local optimization algorithm to calibrate the parameters. In each iteration the software agents perform the following operations: Calculate Fitness, Move, Local Optimization, Evolutionary Operation and Nomination (The agents elect a few individuals with the highest fitness values and age on each planet to form an *Elite Population*). The Evolutionary Operation consists of

- Mutation: The agents model function gets changed randomly: Either a sub-tree of the model function is exchanged or new operations are inserted.
- Crossover: When an agent moves it may happen that the chosen area is already occupied with another individual. In this case, a sub-tree of the individuals model function is replaced by a randomly chosen, suitable sub-tree of the other agents model function.
- Replication: The agent duplicates himself.
- Global Optimization: The agents, which own enough energy or are not adult yet, optimize the parameters of their model function in the Memetic Coprocessor, explained below.

The global optimization is realized on memetic coprocessors, running on an extra processor core, executing more sophisticated algorithms for a global optimization. In the current configuration we use a downhill-simplex algorithm (Nelder and Mead 1965). The algorithm chosen for this local parameter optimization is resource-saving, because it is executed for all agents in every iteration.
The machine learning provides the capability of running the evolutionary algorithm described above on several planets at the same time. If this is the case, some of the areas on each planet get marked as so called beam areas. After each iteration copies of all individuals placed on such an area are send to a randomly

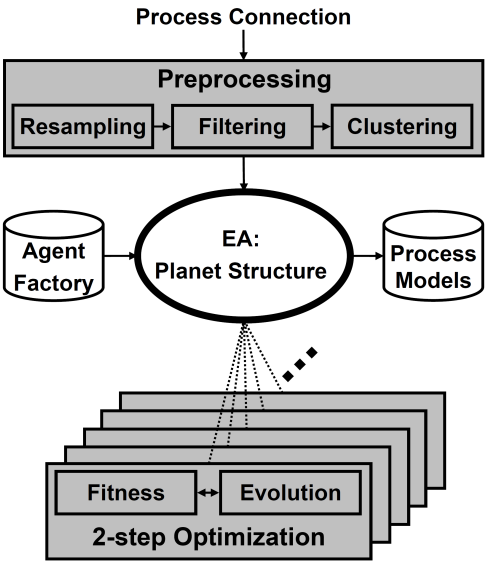

Figure 3: The architeture of the machine learning.

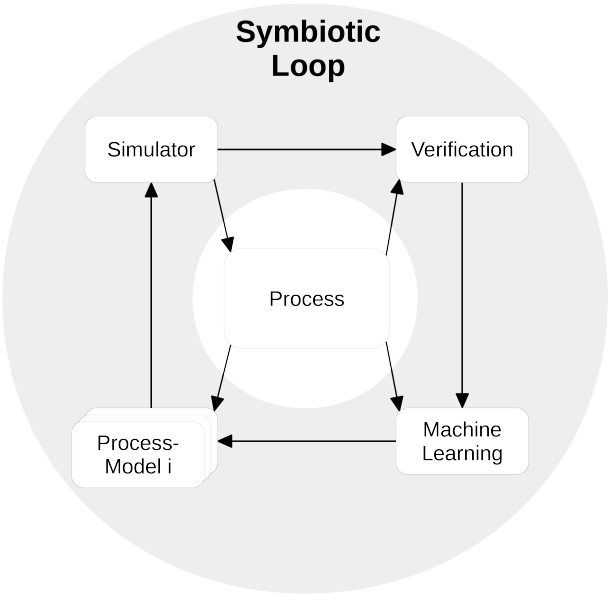

Figure 4: Symbiotic Loop.

chosen area on a randomly chosen planet, provided the chosen area is not yet occupied by an agent. In the experiments 100 of the 6561 areas on every planet were marked as beam areas. Our implementation associates each planet to one processor core, on an additional processor core a universe supervisor is executed. This supervisor manages the elite population using the data from all planets and controls the termination condition. The information exchange between the cores is implemented via a non-blocking Message Passing Interface.

4 VERIFICATION APPROACHES

The verification block confirms the quality of the model and is an essential part, it helps to evaluate the model which is improved or build up during identification process. The verification strategy is based on a set of process input sequences $((x_1)_t, \ldots, (x_m)_t,\, t \in \mathbb{N})$ and output sequences $((y_1)_t, \ldots, (y_j)_t,\, t \in \mathbb{N})$ and of the simulation output sequences $((z_1)_t, \ldots, (z_j)_t,\, t \in \mathbb{N})$. The output sequences of the simulator are related to the input sequences by functional relationships $f\colon \mathbb{R}^m \to \mathbb{R}^j$.In principle, the verification method can be executed with synthetic data (generated by a model of the evaluated system) or live data (acquired from the physical process). We first focus on the synthetic data verification.
The Curve Fitting, introduced in subsection 3.2 provides an open loop verification, which is not able to start the verification process without an initial model. Furthermore, this open loop verification is not able to control the verification process to increase the quality of the model. Precisely for this reason, we have designed a closed-loop verification process, the symbiotic loop, shown in Figure 4.

The system architecture follows the paradigm of the symbiotic loop, shown in figure 4 and it is based on the machine learning, introduced in subsection 3.3. Five modules form this symbiotic cycle which is application independent, the additional modules are described in the following.

Process The process block covers all physical relations of the considered process. Analog inputs or outputs have to be transformed using analog-to-digital- or digital-to-analog-converters. The interface to the digital in-/out-signals is handled using the process data streaming protocol (PDSP) managing distributed process data flows (Bohlmann, Klinger, Szczerbicka, and Becker 2010). This protocol is designed to be used in mixed continuous and discrete environments (Zeigler, Praehofer, and Kim 2000) referred to as hybrid.

Focusing on symbiotic simulation (Fujimoto, Lunceford, Page, and Uhrmacher 2002), PDSP is primary designed to satisfy four modes of operation (analytic, transparent, online, prediction); here we focus on the online mode. In this mode PDSP is used to simulate a process and transmit the results back to the process. The data is directly transmitted between the physical process and the simulator. Therefore latency is minimized although proxy servers may be necessary for large scale simulations.

Simulator In the simulator block we are using two different approaches, dependent from the application. For general purpose it is a Java based simulation system specifically designed for high speed online and symbiotic simulations. The simulator especially has online compiling capabilities, e.g. models can be compiled during run-time in memory and then dynamically injected to the simulator. It is capable to dynamically load or receive models (basically any kind of java program) and simulate multiple isolated instances in the same memory/thread context. The simulator combines Java class loading mechanism and byte code enhancing to calculate user defined metrics while processing prior structural unknown models on the fly. In combination with an OSGI framework PDSP can be directly embedded to running simulations.
For the specific application of ENG-based identification we are using currently the well established NEURON framework for empirically-based simulations of neurons and networks of neurons (Carnevale and Hines 2006), (Coates, Larson-Prior, Wolpert, and Prior 2003), (Law and Kelton 2000). The different constraints, like myelin structures, all-or-none, two directions of information flow, frequency borders of the action potentials, etc. has been taken into account. We have configured the simulator and realized a complex neural bundle including our cuff electrode setup to generate verification data for several information transfer scenarios. The action potentials used for the NEURON-simulator are derived by human arm modeling via Matlab Robotics Toolbox (Corke 2011). With this model for verification we are able to concentrate on specific muscle groups and their reactuatory answer and therefore we are able to generate verification pattern. The simulation environment uses the Hodgkin-Huxley model (Hodgkin and Huxley 1952) to simulate the axon internal membrane, the ion-channels and the extra-cellular space. So, the propagation of action potentials along the axons is modeled using these equations. Furthermore, the mechanisms concerning the passive membrane channels are included.

Process Model The area provides data samples to learn from and calculate the error of the model function ($\mathbb{R}^m \rightarrow \mathbb{R}$) is stored in a tree representation. This function is composed of elementary operations (Schmidt and Lipson 2007), like $+, -, *, /, sin, sqrt$ the variables $x_1, \ldots, x_m$ and a set of parameters within their model function.

According the multi-agent capability, there are several models existing. The number of currently active agents and their inner candidate function is regulated by a software PID control. It is programmed to use the available processing power in nearly optimal conditions. The best models are picked according their ability to survive multiple times longer than the mean agent population. This metric corresponds to the selection of the fittest agents. The complexity of the process models is either dynamic. It depends on the number of input variables, constants and operations. The complexity is defined by counting the number of nodes in the tree representations. The differences between both verification methods are illustrated in Figure 5. The open-loop method allows a verification of the potential-level (electrode-data) against the command-level (Stim-Pattern). It is not able to generate new adequate verification pattern to improve the quality of the model behavior according to specific system states. The new method including the symbiotic loop integrates the verification in a complex and interactive exchange of data between physical process, process model, machine learning and the simulator. The closed-loop verification provides a verification between the trajectory level and the command-level, including the potential-level. The identification block controls the generation of trajectory-related Stim-Pattern.
To create a symbiotic online learning environment all data from the physical process is streamed via PDSP to and from a central routing software. Machine learning and the massive parallel simulation are running outside the roots real time domain. The control simulation, which is using the currently best known model, is running inside an embedded system in real time. Communication between the two domains basically

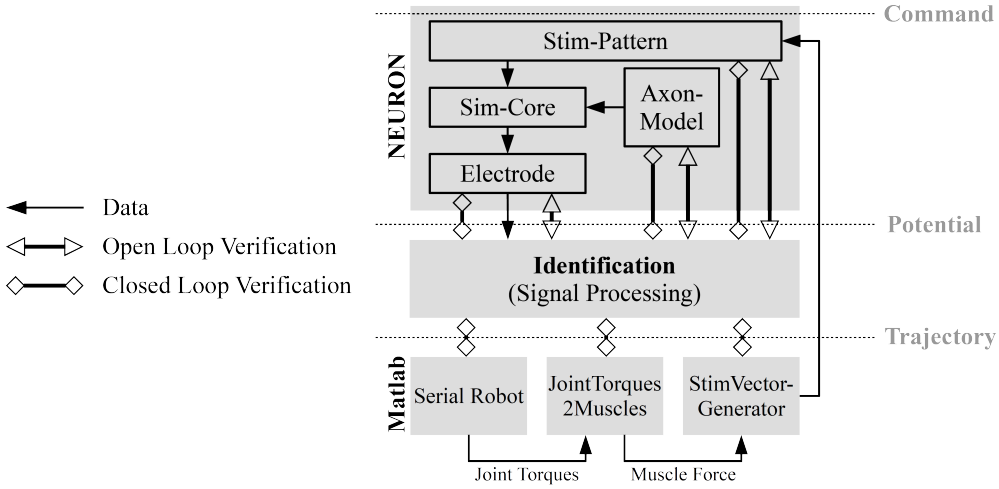

Figure 5: Open- and closed-loop verification.

consists of a model transmission and a movement direction proposal directed into the real time domain. Data transmission directed to the machine learning, verification and simulation is a set of streamed time series data.

5 RESULTS

In order to validate the identification procedure proposed in this paper, we present here one scenario based on the movement of the forearm and the hand where the generation of the Stim-Pattern is focused on the movement of the forearm. The resulting time series of simulated extracellular recordings are fed to the identification algorithm.
The sequence of signals is transformed into actions potentials and simulated by the NEURON simulator using the axon bundle configuration and the cuff electrode setup. The data acquired by this simulated cuff electrode is shown in figure 6. In this figure the frequency of the action potentials and their signal amplitudes, caused by the superposition of several actions potentials, is shown.

Figure 7 shows some details from the overall signal sequence in figure 6. it shows the simulated data for three adjacent rings of the cuff electrode. The three time series are different because of the superposition of actuatory and reactuatory signals during the zoomed time period.

Figure 8 shows a set of Action Potentials (grey, input data from the cuff-electrode) and its approximation (red) executed by the first-level algorithm of the Curve-Fitting method. The time period between 2600-2800 ms shows the superposition of several action potentials which are not in phase. This time period is zoomed in Figure 9 to show the superposition in more detail.
The dashed curves are single action potentials located on different axons. The superposition of these both action potentials forms the red curve based on the action potential characteristics, we call action potential templates (amplitude, frequency, refraction period, etc.).

In the figures 10 and 11 two additional aspects of the identification process are shown. Here the identification method is using only the data acquired by the cuff electrode, no additional information. In Figure 10 the detection of the direction to identify actuatory and reactuatory signals is shown, based on the Stim-Pattern, generated from the Matlab Robotics Toolbox. In Figure 11 the cluster assignment, identifying the actuatory feedback fascicles, are shown.

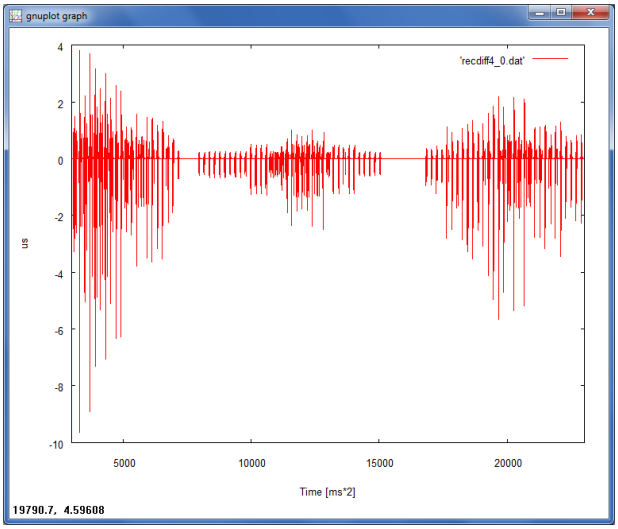

Figure 6: Cuff-electrode data and approximation.

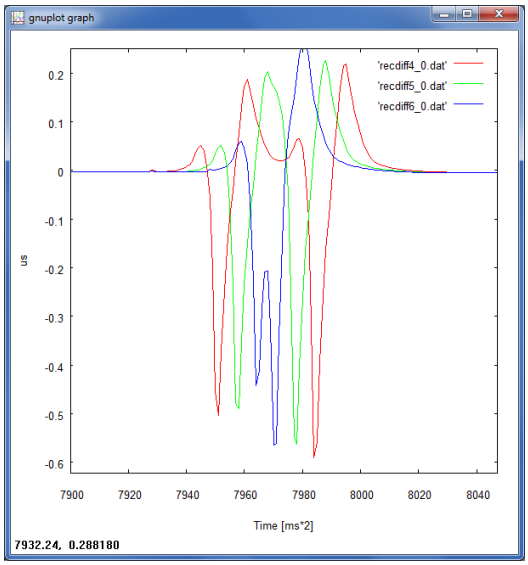

Figure 7: Cuff-electrode data and approximation.

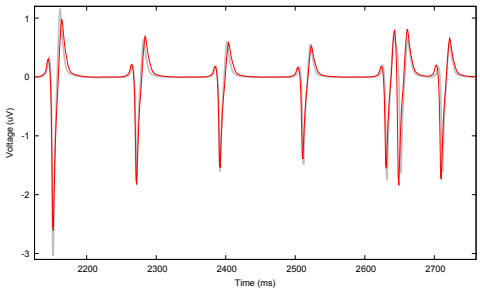

Figure 8: Cuff-electrode data and approximation.

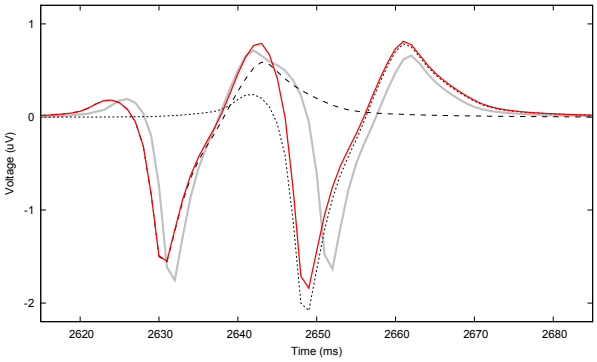

Figure 9: Cuff-electrode data and approximation.

6 SUMMARY AND FURTHER WORK

The SMoBAICS acquisition system motion-based action potentials in neural bundles for exo-prostheses control or for handicapped limb simulation provides a integrated solution from action potential recording up to the identification procedure. This paper focuses on two different verification methods, providing a open- or closed-loop approach. While the open-loop approach helps to identify well-known processes, improving the overall model quality, the open-loop approach, using the symbiotic loop, helps to generate a physical process model from scratch.
A continuous symbiotic system was presented, where different modules, like machine learning, simulator and process models are interacting with a physical or technical process. It improves the identification progress and opens the symbiotic simulation to applications from biotechnology.

The further work has the following key aspects:

- Evaluating the high performance Java simulator instead of the application-specific Neuron simulator.
- Evaluation of the whole identification process, regarding quality and performance.
- Adapting the symbiotic loop to a clinical environment, to replace the Matlab-based part and to integrate the identification into a learning environment and the concerning applications.

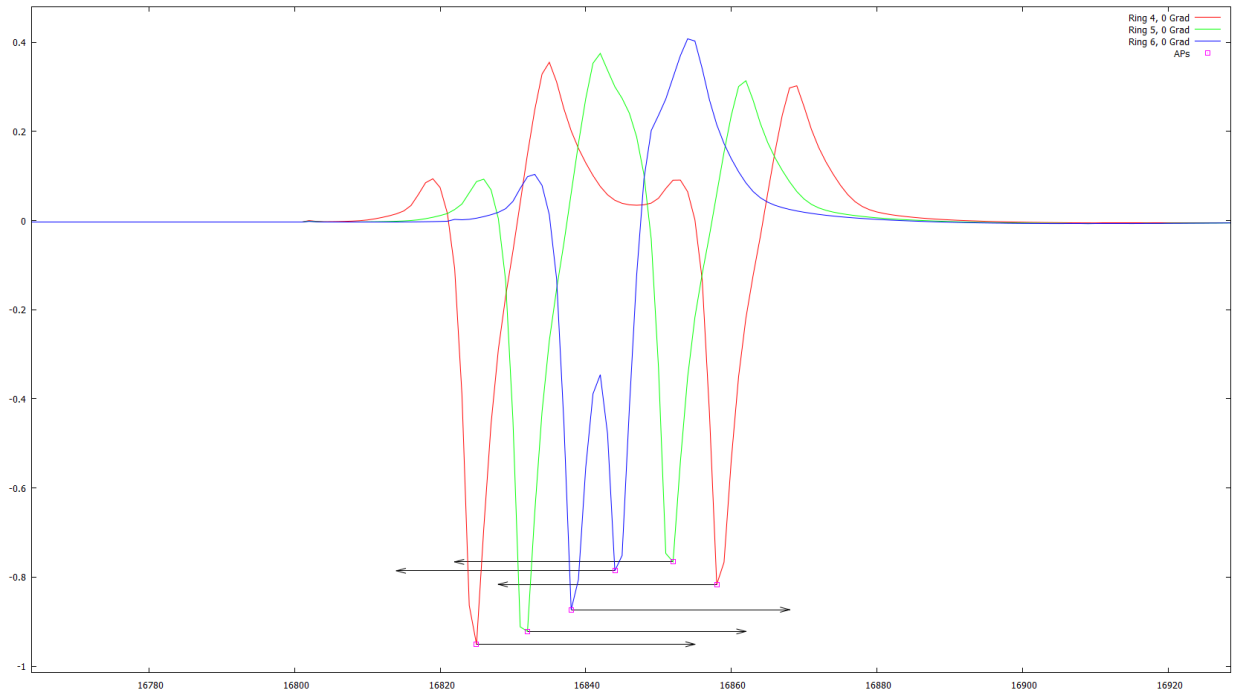

Figure 10: Restructured action potentials using the identification method and detection of actuatory and reactuatory action potentials (x-axis: time index [ms], y-axis: [μV].

7 ACKNOWLEDGEMENT

All in-vivo experiments are performed in collaboration with the Clinic for Neurosurgery at the Hannover Medical School (MHH).

REFERENCES

Aydt, H., S. J. Turner, W. Cai, and M. Y. H. Low. 2008. "Symbiotic Simulation Systems: An Extended Definition Motivated by Symbiosis in Biology". In *Proceedings of the 22nd Workshop on Principles of Advanced and Distributed Simulation*, PADS '08, pp. 109–116. USA, IEEE Computer Society.

Aydt, H., S. J. Turner, W. Cai, and M. Y. H. Low. 2009. "Research Issues in Symbiotic Simulation". In *Proceedings of the 2009 Winter Simulation Conference M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, eds.*

Bohlmann, S., A. Klauke, and V. Klinger. 2013. "In-vivo-Experiment zur System-Validation des modularen Mess-Systems NGEPM-2". *Appear June 2013: Forschungsberichte der FHDW Hannover (ISSN 1863-7043),* vol. 02.

Bohlmann, S., A. Klauke, V. Klinger, and H. Szczerbicka. 2011, September. "Model Synthesis Using a Multi-Agent Learning Strategy". In *The 23rd European Modeling & Simulation Symposium (Simulation in Industry)*. Rome, Italy.

Bohlmann, S., V. Klinger, and H. Szczerbicka. 2009. "HPNS - a Hybrid Process Net Simulation Environment Executing Online Dynamic Models of Industrial Manufacturing Systems". In *Proceedings of the 2009 Winter Simulation Conference Rossetti, Hill, Johansson, Dunkin, and Ingalls, eds.*

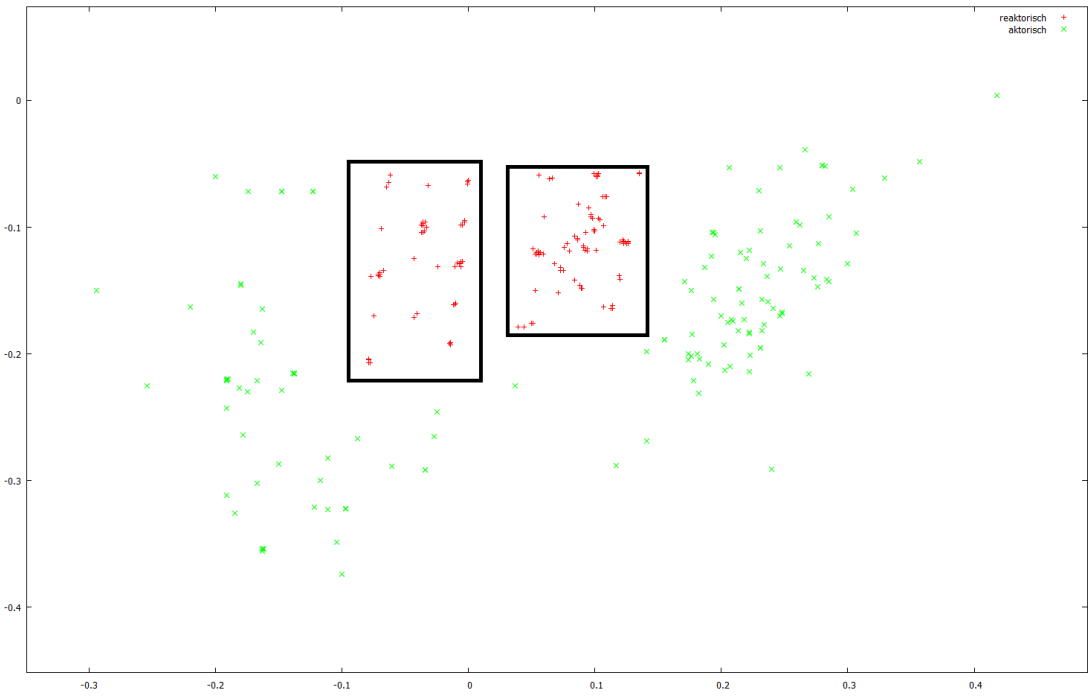

Figure 11: Clustering of actuatory and reactuatory axons within axon bundle (xy-area: localization of axons).

Bohlmann, S., V. Klinger, and H. Szczerbicka. 2010. "System Identification with Multi-Agent-based Evolutionary Computation Using a Local Optimization Kernel". In *The Ninth International Conference on Machine Learning and Applications*, pp. 840–845.

Bohlmann, S., V. Klinger, H. Szczerbicka, and M. Becker. 2010, June. "A Data Management Framework Providing Online-Connectivity In Symbiotic Simulation". In *24th EUROPEAN Conference on Modelling and Simulation, Simulation meets Global Challenges*. Kuala Lumpur, Malaysia.

Carnevale, N. T., and M. L. Hines. 2006. *The NEURON Book*. New York, NY, USA, Cambridge Univ. Press.

Cesqui, B., P. Tropea, S. Micera, and H. Krebs. 2013. "EMG-based pattern recognition approach in post stroke robot-aided rehabilitation: a feasibility study". *Journal of NeuroEngineering and Rehabilitation* vol. 10 (1).

Coates, T.D., J., L. Larson-Prior, S. Wolpert, and F. Prior. 2003. "Classification of Simple Stimuli Based on Detected Nerve Activity". vol. 22 (1), pp. 64–76.

Corke, P. 2011. *Robotics, Vision and Control - Fundamental Algorithms in MATLAB®*, Volume 73 of *Springer Tracts in Advanced Robotics*. Springer.

De Jong, K. A. 2006. *Evolutionary Computation: A Unified Approach*. MIT press.

Fujimoto, R., D. Lunceford, E. Page, and A. Uhrmacher. 2002. "Technical Report of the Dagstuhl-Seminar Grand Challenges for Modelling and Simulation". Technical Report No. 02351, Schloss Dagstuhl, Leibniz-Zentrum für Informatik, Wadern, Germany.

Hazan, L., M. Zugaro, and G. Buzsáki. 2006, March. "Klusters, NeuroScope, NDManager: A free software suite for neurophysiological data processing and visualization.". *J Neurosci Methods* (155), pp. 207–216.

Hodgkin, A., and A. Huxley. 1952. "A quantitative description of membrane current and its application to conduction and excitation in nerve". *Journal of Physiology* vol. 117, pp. 500–544.

Klinger, V. 2015, September. "Biosignal Acquisition System for Prosthesis Control and Rehabilitation Monitoring". In *4nd International Workshop on Innovative Simulation for Health Care (IWISH 2015)*, edited by A. Bruzzone, M. Frascio, V. Novak, F. Longo, Y. Merkuryev, and V. Novak.

Klinger, V., and A. Klauke. 2013. "Identification of motion-based action potentials in neural bundles using an algorithm with multiagent technology". In *2nd International Workshop on Innovative Simulation for Health Care (IWISH 2013)*, edited by W. Backfrieder, M. Frascio, V. Novak, A. Bruzzone, and F. Longo.

Law, A. M., and W. D. Kelton. 2000. *Simulation Modeling and Analysis*. McGraw-Hill.

Micera, S., J. Carpaneto, and S. Raspopovic. 2010, October. "Control of hand prostheses using peripheral information". *IEEE Reviews in Biomedical Engineering* vol. 3, pp. 48–68.

Micera, S., L. Citi, J. Rigosa, J. Carpaneto, S. Raspopovic, G. D. Pino, L. Rossini, K. Yoshida, L. Denaro, P. Dario, and P. M. Rossini. 2010. "Decoding Information From Neural Signals Recorded Using Intraneural Electrodes: Toward the Development of a Neurocontrolled Hand Prosthesis.". *Proceedings of the IEEE* vol. 98 (3), pp. 407–417.

Nelder, R., and J. Mead. 1965. "A simplex method for function minimization". *Computer Journal* vol. 7(4), pp. 308–313.

Schmidt, M., and H. Lipson. 2007. "Comparison of tree and graph encodings as function of problem complexity". In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pp. 1674–1679. New York, NY, USA, ACM.

Zeigler, B. P., H. Praehofer, and T. G. Kim. 2000. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. 2 ed. San Diego, USA, Academic Press.

AUTHOR BIOGRAPHIES

SEBASTIAN BOHLMANN is a Ph.D. candidate at Department of Simulation and Modeling - Institute of Systems Engineering at the Gottfried Wilhelm Leibniz Universität Hannover. He received a Dipl.-Ing. (FH) degree in mechatronics engineering from FHDW university of applied science. His research interests are machine learning and heuristic optimization algorithms, complex dynamic systems, control system synthesis and grid computing. His email address is <bohlmann@ifam.uni-hannover.de>.

VOLKHARD KLINGER is a professor for embedded systems and computer science at the university of applied science FHDW in Hannover and Celle since 2002. After his academic studies at the RWTH Aachen he received his Ph.D. in Electrical Engineering from Technische Universität Hamburg-Harburg. During his 8-year research activity at the Technische Universität Hamburg-Harburg and research laboratories he did research in asic design and automation. He teaches courses in computer science, embedded systems, electrical engineering and ASIC/system design. His email address is <Volkhard.Klinger@fhdw.de>.

HELENA SZCZERBICKA is head of the Department of Simulation and Modeling - Institute of Systems Engineering at the Gottfried Wilhelm Leibniz Universität Hannover. She received her Ph.D. in Engineering and her M.S in Applied Mathematics from the Warsaw University of Technology, Poland. Dr. Szczerbicka was formerly Professor for Computer Science at the University of Bremen, Germany. She teaches courses in discrete-event simulation, modeling methodology, queuing theory, stochastic Petri Nets, distributed simulation, computer organization and computer architecture. Her email address is <hsz@sim.uni-hannover.de>.