Towards bioinspired close-loop local motor control: a simulated approach supporting neuromorphic implementations

Fernando Perez-Peña,
J. Antonio Leñero-Bardallo
School of Engineering,
University of Cadiz, Spain
fernandoperez.pena@uca.es

Alejandro Linares-Barranco
Robotic and Technology of
Computers Lab,
School of Computer Engineering
University of Seville, Spain

Elisabetta Chicca
Cognitive Interaction Technology,
Center of Excellence (CITEC)
Faculty of Technology
Bielefeld University, Germany

***Abstract*—Despite being well established in robotics, classical motor controllers have several disadvantages: they pose a high computational load, therefore requiring powerful devices, they are not easy to tune and they are not suited for neuroprosthetics. In contrast, bio-inspired controller do not transform the output of the controller therefore no delays are introduced and a smooth response is achieved; they also have a high scalability. Finally, the most important feature of bio-inspired controllers is that they could integrate learning features to make them adaptable to new tasks within the same hardware robotic platform. We present the model and simulation of a spiking neural network for low-level motor control. The proposed neural network acts as a motor controller and produces pulsed signals which can be directly interfaced with commercial DC motors. The simulated network is compatible with neuromorphic VLSI implementation and paves the way to the implementation bio-inspired motor controller which are compact, low power, scalable and compatible with neuroprosthetic. The network presented is inspired by the current knowledge about biological motor control: it comprises alpha motoneuron for driving the motor and spindle populations to provide the feedback and close the loop. The spikes from the motoneuron population are time lengthen to a fixed amount of time and supplied to the simulated motor: Pulse Frequency Modulation (PFM) modulation is used. This paper presents the software simulations using the Brian simulator for a position controller. Our controller is a first step toward a novel bio-inspired motor control approach suitable for robotics as well as neuroprosthetic.**

Keywords*—motor control, PFM, motoneurons, neuromorphic engineering, robotics.

I. Introduction

Conventional motor controllers range from very simple proportional actions to complex fuzzy logic or chaotic controllers. They are designed to work under specific constraints and environments. In contrast, biological motor control exhibits amazing flexibility; for instance, a group of muscles can be trained to do any task. In particular, the last stage of muscle control, namely local motor control or short-loop, is well understood and can be imitated for designing efficient control architectures.

The classical approach to motor control focuses on techniques aimed at producing precise movements. These control techniques are general and they can be adapted depending on how the command and the feedback are supplied, known as: feedforward, feedback or adaptive control. Alternatively, the knowledge about biological structures involved in the generation of movements provided by neuroscience can be exploited in the form of models and Neuromorphic Engineering (NE) implementations to develop novel approaches to artificial motor control. The NE community has been already quite active in integrating sensors and computational devices with several robotic platforms (significative examples are [1], [2]). In comparison, very little work has been carried out so far for the development of neuromorphic motor controllers.

To our knowledge, the first neuromorphic controller was proposed by Fukuda and colleagues [3]. They proposed a hierarchical neural network controller applied to a robotic manipulator. The model was based on the neuron's firing rates, therefore ignoring the role of spikes. We propose to use the spikes to directly drive the motor, therefore removing delays on the execution and reducing computational load. This first attempt did not trigger much research in the NE community and only many years later new publications presenting neuromorphic motor controllers appeared. For example, a Dynamic Vision Sensor (DVS) has been used as visual input or as a feedback provider [4] [5] where the tasks were to balance a pencil and to generate pointing movements. Further work introduced the use of Proportional Integral and Derivative (PID) controllers based on small network of spiking neurons implemented on Field Programmable Gate Arrays (FPGAs) or SpiNNaker [6]–[8]. However, this controllers were developed under industrial constrains that might not fit the neuromorphic engineering goals. More recent work presented in [9] and [10] describes the use of neuromorphic hardware to control the robot motor torques and a small robotic arm, respectively. The differences between the present work with the former is that we are considering a single motor instead of a whole system and with the latter is that we are considering the biological features of the spinal cord instead of the basal ganglia. In the latest work in this field a brain-like neural controller including a cerebellum model and a musculoskeletal robot were presented [11]. Dedicated computational cores were used to translate the spike trains to motor commands. In our work this computational stage is replaced with a direct interface of the spike train with the Direct Current (DC) motor.

Given our aim of replicating and exploiting biological strategies, a spike based implementation is a natural choice. This paper focuses on motor control and specifically on how we can use neuromorphic hardware to control a motor following a biological approach. The objective of the work presented here is to design a new low level control method based on biological features that could be used in robotics. Our long term plan aims at the integration of NE hardware with DC motors. To this end we simulated Leaky Integrate and Fire (LIF) neurons interconnected through excitatory and inhibitory synapses emulating biological dynamic properties such as an instantaneous rise and single-exponential decay and used the output of our network model to drive a simulated DC motor using its transfer function to relate voltage applied versus angle reached over time.

In this paper we describe the biological background and the methodology followed II-A, the network model proposed (II-B), the results for the software simulations of the control model (III), and finally discuss current achievements (IV).

II. Methodology

A. What can we borrow from biology?

Since we want to use the control method for robotics, we have to focus on motors instead of muscles and that is the reason why we do not consider key aspects of muscles such as elastic properties, agonist-antagonist mechanism, etc. In contrast, motors do not only pull but push just by inverting the polarity of the voltage applied to its terminals. A key point in biology is how the proprioceptive information related to the muscles is transmitted to the central nervous system: using the spindles (to provide length and velocity data) and the Golgi tendon organs (to provide tension data) [12]. Our objective is to control either the position or the speed therefore we are going to focus on how spindles work. In general terms, spindles are neurons placed along the intrafusal fibres that respond to the changes in length of the muscle. Their output is a constant rate proportional to the muscle length [12]. In a similar way we are going to use the encoder of a DC motor to generate a spiking signal proportional to the position or velocity of the motor to provide feedback to the neural controller. Regarding the actuation, the stretching of the fibres of a muscle happens when the central nervous system recruits the neurons called alpha motoneurons [12]. The amount of strength depends on the discharge to these motoneurons. Our approach mimics this behaviour.

B. Network model

The designed network is shown in figure 1. We propose four different neuron populations: two populations to drive the motor in each direction (clockwise and counter clockwise) and two populations to provide feedback from the motor to the neural controller. The alpha motoneurons CW and alpha motoneurons CCW output spikes are directly connected to the motor and drive CW and CWW movements respectively. The motor encoder's output consist of two square signals which are out of phase (channel A and B) and it is provided to the spindle populations. The reference is shared by both alpha motoneurons populations and the input *Direction* selects what type of connection is created between the reference and

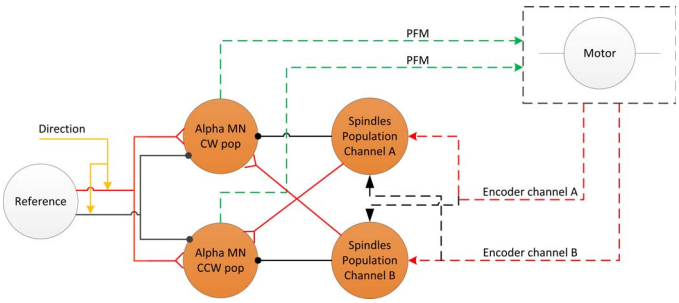


Fig. 1. Control network diagram. The dotted arrows represent connections between neuron populations and the motor and the solid arrows connections between neuron populations. Orange-filled circles represent dynamic populations, inhibitory connections in black and excitatory connections in red. The stimulus to the network is supplied using the reference population and the input direction selects the type of connection, between the stimulus and the alpha motoneuron populations, to be created.

the motoneuron populatios: excitation for the clockwise motoneuron and inhibition for the counter-clockwise motoneuron, whenever there is a clockwise movement and the contrary for counter-clockwise movements. The spindles populations are excited with the turn of the motor in each direction (channel A for clockwise and channel B for counter-clockwise). The four neuron populations are modelled using the LIF model 1

$$\tau_m * \frac{du}{dt} = -(u - R * (I(t) + I_0)) \quad (1)$$

where τ_m is the neuron membrane time constant, u is the membrane potential, $I(t)$ is the dynamic injected current and I_0 is a constant current that could be injected. The model fires a spike whenever the threshold is reached and then the membrane potential is reset. The reference population provides input pulses at a rate proportional to the target position (in degrees) and the direction input specifies the direction of the turn by determining the sign of the connection from the reference to the alpha motorneurons. During a clockwise turn command, the reference excites the CW population and inhibits the CCW population; the opposite occurs for the counter clockwise direction. Within this mechanism, we make our controller flexible since the turn direction can be selected and changed online. As soon as the motor starts turning, the spindle populations are stimulated by the encoder signal; eventually they will start firing and inhibiting the corresponding alpha motoneuron population, therefore stopping the turning of the motor. If, due to motor inertia, the position is overreached, the cross connection between the encoder populations and the alpha motoneurons will correct the position by exciting the opposite motoneuron population. Each population comprises 30 neurons with local recurrent random connections. These connections introduce variability in the response which is integrated by upstream neurons to produce a smooth response over time.

C. Motor driver

Our approach uses directly the output rate from the alpha motoneuron populations to drive the motor, i.e. the information we are sending to the motor (the spikes) is encoded in the firing rate. Therefore, we have used a modulation based on the frequency: PFM. PFM uses a squared waveform to carry the information. The pulse width is fixed and the analog

information is carried by the frequency. This means that the frequency of the square wave will be modified according to the analog value to transmit. The amount of time when all the power is supplied to the load is fixed [6]. The idea of using the frequency as the information carrier links precisely with using the firing rate of the alpha motoneurons to drive the motor. If we compare our approach with the well-known Pulse Width Modulation (PWM), we found that PWM uses a fixed frequency squared waveform to carry the analog value. The modulator will spread each pulse according to the analog value, introducing a delay. Thus, the width of each pulse will have a relationship with the analog value to transmit. Therefore, the period of time when all the power is supplied to the load is variable. The use of PFM is validated by the interest of those working on neuroprosthetic [13] because it is the most natural way to interface spikes coming from biological neurons. The motor we plan to use in the final system is a DC Motor from Maxon (Model 310007). Following the specifications of the company, the dynamic properties of the motor can be modelled using its differential equations 2 [14]. The motor model is included within the simulator by using a fake neuron that follows these equations.

$$V_s = L_a \times \frac{di_a}{dt} + R_a \times i_a + k_e \times \frac{d\theta}{dt} \tag{2}$$

$$k_t \times i_a = J \times \frac{d^2\theta}{dt^2} + B \times \frac{d\theta}{dt} \tag{3}$$

where V_s and i_a are the voltage and current applied to the motor terminals, θ is the angle reached by the motor and L_a, R_a, k_e, k_t, J and B are the motor parameters given by the manufacturer. In the eventual hardware implementation, the output of the alpha motoneurons will use a dedicated digital bus to interface the digital circuit that will generate the PFM signal to drive the motor with the right polarity according to the turning direction.

D. Feedback

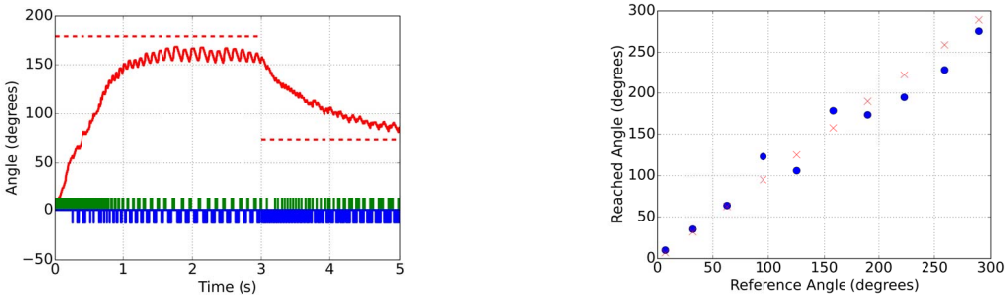
A feed-forward controller relies on the execution of the motor command assuming that the target was reached after the predicted time. This approach is not robust to interferences and cannot be used for moving targets with unpredictable trajectories. In these cases feedback plays a crucial role. In our controller, low-level feedback is introduced by using the output of the encoder included within the DC motor which is analogous to biological proprioceptive signals. Specifically, the encoder currently in use (Maxon MR225778) is an incremental encoder with three output channels (two quadrature output channels plus the index channel to detect a full turn). The two quadrature channels will be interfaced using a digital circuit (eventually implemented on an FPGA). The circuit could either generate pulses according to the switching frequency of the channels (rate proportional to the speed of the motor) or generate a rate according to the number of pulses received from channel A or B (rate proportional to the position of the motor). Depending on how we want to control the motor, one of these firing rates will excite the encoder neuron population implemented on the analog chip. In the simulations we present in this paper, we have designed a position control network. Therefore the spindle populations (interfaced by the encoder channels) receive pulses at a rate proportional to the position of the motor. Equation 1 is a first approximation to the analog neuron circuit implemented in the chips [15] we plan to use in our hardware implementation. With no input current (no turn by the motor), the spindle population will not be firing and with each spike received by the encoder (encoding 0.72 degrees each), an increasing firing rate will be generated at its output. The firing rate generated by this block follows the equation 4. This behaviour mimics the response of biological spindle neurons to muscle length variations. Given a fix set of parameters (τ_m, R, I_0 and the threshold) the output rate will depend on the input current. This output rate will inhibit the corresponding alpha motoneuron population which is receiving excitation from the reference. This reference stimulus is the input to our network. Eventually, it will be set by higher structures in the hierarchy beyond the scope of this work.

III. RESULTS

The network designed has been simulated using the spiking neural network simulator Brian [16]. The simulations include all the elements shown in 1: the four neuron populations and two equation blocks for the motor function and the encoder's channels dynamics.Firstly, we tune the connection between the encoder channels and the spindle populations. To be consistent with both the biological spindle (which rate ranges from 0 to $100s^{-1}$ [17]) and the computational limitation of having a significant rate variation at each reference step, we fixed the parameters with the values: $\tau_m = 25.4ms$, $R = 10\Omega$, $I_0 = 2mA$ and the $u_{th} = 20mV$. With these values, the equation 4 is used to compute the equivalent of the firing rate generated by the spindle population. The encoder has a resolution of 500 pulses per full turn, i.e. one pulse every 0.72 degrees, which fixes the best achievable resolution of the system. Hence, each time the motor turns 0.72 degrees, the equation block used to include the motor in our simulations fires a spike. This spike excites the corresponding spindle population. Therefore, the spindle population firing rate is proportional to the encoder resolution.

$$f_{encoder} = 1/\left(\tau \times ln\left(\frac{R \times \left(\frac{Angle}{1440} + I(t)\right)}{R \times \left(\frac{Angle}{1440} + I(t)\right) - V_{th}}\right)\right) \tag{4}$$

Directly using the highest resolution would lead to a spindles firing rate much higher than the biological one. Therefore, we have artificially increased the step size three times, up to 2.16 degrees which means 166 steps for a full turn of the motor. Every time the encoder fires a spike, the current injected to the spindle population is increased by $50uA$. We run tests to study how the system responds to a range of pulse width of the PFM signal which drives the motor. The position reached will depend on both the pulse width and the amplitude. For this specific motor, we have fixed the amplitude to 12 Volts according to the manufacturer datasheet. The tests show the expected linear relation between the pulse width and the angle reached: $Angle = 0.02 \times pulse\_width$ with the angle in degrees and the pulse width in μs. We have selected a pulse width of 100 μs (3.3 μs for each the 30 neurons in the population) for the PFM signal to drive the motor. Figure 2a shows the entire controller behaviour. The reference for this test is set to 100 spikes per second during 3 seconds and then 50 spikes per second during 2 seconds



(a) Test I (b) Test II

Fig. 2. Test I: Evolution over time of the position of the motor in response to an input reference of 100 Hz (turn command of 179.28 degrees) for 3 seconds and of 50 Hz (turn command of 73.44 degrees) for the following two seconds without changing the direction input. The input PFM signal for the CW alpha motoneuron population is shown in green and the PFM signal for the CCW alpha motoneuron population is shown in value. Test 2: Angle reached by the motor (10.16, 35.56, 63.78, 123.19, 106.20, 178.52, 173.53, 194.91, 228.7 and 275.73) for the following set of angle references (7, 32.4, 62.64, 95.04, 125.28, 157.68, 190, 222.48, 259.2 and 289.44) degrees. The experiment lasts five seconds each.

(5 seconds total); for both, the clockwise direction is active. The cross connection between the spindle and the opposite alpha motoneuron population are causing both the oscillation around the set reference and the possibility of updating the reaching point when the reference changed. According our initial calibration procedure, the motor should reach 179.28 degrees (before the reference change) but, as shown in the figure, it reaches 172.05 degrees. The error of 4.35% is due to random variations in the inhibition from the spindle population to the alpha motor neuron population. To confirm this, we run 10 simulations restarting the network each time for the first three seconds and obtained a mean error of: 17.45 degrees (10.56%) and a standard deviation of 2.92 degrees. Once the inhibition starts having effect, it inhibits a random number of neurons of the motoneuron population while the rest of the neurons will continue making the motor turn. Figure 2b shows the behaviour of the motor controller when a set of references are set.

IV. Discussion and Conclusion

Our simulations show that it is possible to create a network of LIF neurons to control a DC motor. The proposed network exploits some of the features present in the biological counterpart and can be applied to robotic platforms with several degrees of freedom. Our results show that the proposed controller can achieve an accurate behaviour reaching the target position with an mean percentage error of 10.56%. This work is an important step towards our long term goal of building a neuromorphic controlled robotic arm able to perform reliable and adaptable reach movements. To this end, we designed the simulation to be compatible with our hardware and we will use this results to guide the hardware implementation.

Acknowledgment

This work was supported by the Spanish grant (with support from the European Regional Development Fund) COFNET (TEC2016-77785-P) and the DFG funded Excellence Cluster 227 (CITEC, University of Bielefeld).

References

[1] C. Bartolozzi, F. Rea *et al.*, "Embedded neuromorphic vision for humanoid robots," in *CVPR 2011 WORKSHOPS*. IEEE, 2011, pp. 129–135.

[2] T. Delbruck and M. Lang, "Robotic goalie with 3 ms reaction time at 4% cpu load using event-based dynamic vision sensor," *Neuromorphic Engineering Systems and Applications*, p. 16, 2015.

[3] T. Fukuda, T. Shibata *et al.*, "Neuromorphic control for robotic manipulator," in *Engineering Systems with Intelligence*, ser. Microprocessor-Based and Intelligent Systems Engineering, S. Tzafestas, Ed. Springer Netherlands, 1991, vol. 9, pp. 197–204.

[4] J. Conradt, M. Cook *et al.*, "A pencil balancing robot using a pair of aer dynamic vision sensors," in *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*, May 2009, pp. 781–784.

[5] F. Perez-Peña, A. Morgado-Estevez *et al.*, "Neuro-inspired spike-based motion: from dynamic vision sensor to robot motor open-loop control through spike-vite," *Sensors*, vol. 13, no. 11, pp. 15 805–15 832, 2013.

[6] A. Jimenez-Fernandez, G. Jimenez-Moreno *et al.*, "A neuro-inspired spike-based PID motor controller for multi-motor robots with low cost FPGAs," *Sensors*, vol. 12, no. 4, pp. 3831–3856, 2012.

[7] F. Galluppi, C. Denk *et al.*, "Event-based neural computing on an autonomous mobile platform," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014, pp. 2862–2867.

[8] A. Webb, S. Davies, and D. Lester, "Spiking neural PID controllers," in *Neural Information Processing*, ser. Lecture Notes in Computer Science, B.-L. Lu, L. Zhang, and J. Kwok, Eds., vol. 7064. Springer Berlin Heidelberg, 2011, pp. 259–267.

[9] S. Menon, S. Fok *et al.*, "Controlling articulated robots in task-space with spiking silicon neurons," in *Biomedical Robotics and Biomechatronics (2014 5th IEEE RAS EMBS International Conference on*, 2014, pp. 181–186.

[10] F. Perez-Peña, A. Morgado-Estevez *et al.*, "Spike-based VITE control with dynamic vision sensor applied to an arm robot," in *Circuits and Systems (ISCAS), 2014 IEEE International Symposium on*, 2014, pp. 463–466.

[11] C. Richter, S. Jentzsch *et al.*, "Scalability in neural control of musculoskeletal robots," *arXiv preprint arXiv:1601.04862*, 2016.

[12] M. L. Latash, *Fundamentals of Motor Control*, 1st ed. Elsevier, 2012.

[13] J. J. Abbott and S. G. Meek, "Digital emulation of pulse frequency modulation for neuroprosthetic sensory feedback," *IEEE Transactions on neural systems and rehabilitation engineering*, vol. 15, no. 1, pp. 131–135, 2007.

[14] K. Ogata and Y. Yang, *Modern control engineering*, 5th ed. Prentice Hall, 2009.

[15] E. Chicca, F. Stefanini *et al.*, "Neuromorphic electronic circuits for building autonomous cognitive systems," *Proceedings of the IEEE*, pp. 1–22, 2014.

[16] D. F. Goodman and R. Brette, "The brian simulator," *Frontiers in neuroscience*, vol. 3, p. 26, 2009.

[17] A. Prochazka and M. Gorassini, "Models of ensemble firing of muscle spindle afferents recorded during normal locomotion in cats," *The Journal of physiology*, vol. 507, no. 1, pp. 277–291, 1998.