Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

A model of motor learning in closed-loop brain-machine interfaces: predicting neural tuning changes

Rodolphe Héliot[1,3], Jose M. Carmena[1,2,3]
[1] Department of Electrical Engineering and Computer Sciences
[2] Program in Cognitive Science
[3] Helen Wills Neuroscience Institute
University of California, Berkeley, CA 94720, USA
E-MAIL: carmena@eecs.berkeley.edu

***Abstract*— This paper presents a model of the learning process occurring during operation of a closed-loop brain-machine interface (BMI). The learning model updates neuron firing properties based on a feedback-error learning scheme, featuring feedforward and feedback controllers. Our goal is to replicate in simulation experimental results showing functional reorganization of neuronal ensembles during BMI experiments. We show that the proposed model can simulate motor learning, and that the predicted changes in neuronal tuning are consistent with experimental observations. We believe that being able to simulate motor learning in a BMI context will allow designing decoders that would facilitate the learning process in real world experiments.**

***Keywords*— Brain-Machine Interfaces - Motor learning – Directional tuning**

I. INTRODUCTION

A BMI is a direct connection between the brain and an artificial system. Motor BMIs aim to control external devices such as robotic arms or computer generated cursors. In a cortical BMI system, recorded signals from the brain are fed into a mathematical algorithm which translates these signals into a motor plan, i.e. the subject's intention of movement. This motor plan is then streamed to an artificial actuator, the prosthetic arm. A closed control loop is established by providing the subject with visual feedback of the state of the prosthetic device (Figure 1). BMI research has led to demonstrations of non-human primates and humans controlling prosthetic devices in real-time through modulation of neural signals [1-8].

A critical component of a BMI system is the decoder that transforms the neural activity into a motor plan. Such decoders are often built by recording neural data together with behavioral data when subjects are performing a reaching task. Decoder parameters are then set to obtain good predictions of the behavioral data (e.g. position) based on the neural recordings. This decoder can be then used in a real-time closed-loop BMI experiment.

Several BMI studies have reported functional changes in the neurons involved in closed-loop BMI control [1,2,7,8,9]. In particular, these studies have shown that improvements in performance require learning and are associated with changes in neuronal tuning properties. Indeed, to successfully control the prosthetic device in a BMI paradigm, the brain has to learn the transform that converts neural activity into a motor plan. From a motor control point of view, learning a specific motor transformation implies building an inverse model of the decoder [10].

A recent study by Jarosiewicz et al. [9] reported functional reorganization of neuronal tuning properties after perturbation of the learned decoder. In this study, after generating a decoder from baseline neural activity, the authors perturbed the decoder by modifying its tuning for a subset of the recorded cells. They made several observations: first, there is a global change in neurons' tuning that compensates for the error in cursor movement. Second, the perturbed set of neurons experienced larger compensatory changes in tuning than the unperturbed set.

This paper investigates the functional reorganization of neurons involved in closed-loop BMI control. To achieve this, we propose a model of motor learning in a BMI paradigm. We would like to replicate in simulations the experimental results showing functional reorganization. We believe that building such a model of motor learning in a BMI paradigm can give us detailed insights of the underlying processes, since the motor transformation in a BMI paradigm is exactly known.

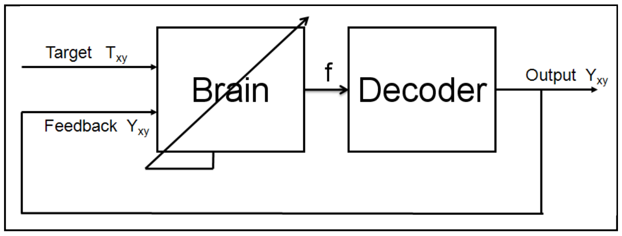

Figure 1. Motor learning in closed-loop Brain-Machine Interface paradigm. The firing rates *f* of the recorded cells are used as inputs by the decoding algorithms, these are converted into a motor plan (output). The brain adapts its firing rates based on the target to be reached, and on the BMI output.

II. THE LEARNING MODEL

A. The BMI paradigm

In this experimental paradigm, neuronal firing rates are recorded in real time; the role of the decoding algorithm is to transform this neural data into a motor plan. Let's assume for the present that the velocity of the artificial device is estimated from the recorded cells' firing rate. For simplicity purposes, and without any loss of generality, we constrain the task to a planar environment, hence with two degrees of freedom. The Population Vector Algorithm (PVA) is used to decode the firing rates. It assumes that each neuron is cosine-tuned to the intended movement direction [11]:

$$f = b_0 + m\cos(\theta) \qquad (1)$$

where f is the firing rate of the model neuron, b_0 its baseline firing rate, m is its modulation depth, and θ is the angle between the direction of intended movement and the unit's preferred direction (PD). The cosine tuning fit of each neuron is calculated by regressing the firing rate of the unit to the target direction:

$$f_{ki} = \beta_0 + \beta_{xi}.t_{kx} + \beta_{yi}.t_{ky} + \varepsilon_{ki} \qquad (2)$$

Where f_{ki} is the firing rate of neuron i during trial k, ε_{ki} is noise, $T = (t_{kx}, t_{ky})$ is the unit-vector pointing in the direction of target during trial k. The baseline b_{0i} is identified as β_{0i}, the modulation depth m_i as the magnitude of the vector $\vec{\beta}_i = (\beta_{xi}, \beta_{yi})$, and preferred direction as $\overrightarrow{PD}_i = \vec{\beta}_i / m_i$.

Once the baseline firing rate, modulation depth, and preferred direction have been identified for each neuron, a PVA decoder can be built. First, firing rates are converted to normalized rates through the equation:

$$r_i = \frac{f_i(t) - b_{0i}}{m_i} \qquad (3)$$

Theses normalized rates are then used to compute cursor velocity $V_{xy}(t)$ via:

$$\vec{V}_{xy}(t) = K.\sum_{i=1}^{N} r_i(t).\vec{p}_i \qquad (4)$$

Where K is a speed factor, N is the number of units used for decoding, and $\vec{p}_i$ is the preferred direction of unit i.

Finally, the cursor position Y_{xy} can be updated as:

$$\vec{Y}_{xy}(t) = \vec{Y}_{xy}(t - \Delta t) + \Delta t.\vec{V}_{xy}(t) \qquad (5)$$

B. Simulating learning within a set of motor neurons

Our underlying assumption is that the brain performs information encoding based on its inputs:

$$f = B\left(T_{xy}, Y_{xy}, \lambda\right), \qquad (6)$$

where T_{xy} represents the target to be reached and λ is a set of adaptable parameters internal to the brain. One can view them as parameters of the internal model built by the brain. Equation 6 is a generalization of equation 1, where λ gathers the b_0, m, and $\overrightarrow{PD}$ parameters for all cells.

As seen in previous section, the decoder transforms neural data into a motor plan:

$$\vec{V}_{xy} = PVA(f) \qquad (7)$$

We can assume that the direction of intended movement is the direction pointing from the actual position to the target:

$$\vec{D}_{xy}(t) = T_{xy}(t) - \vec{Y}_{xy}(t) \qquad (8)$$

The cursor movement error $\vec{E}_{xy}$ can be defined as the difference between the direction of the decoded velocity $\vec{v}$ and the intended direction:

$$\vec{E}_{xy} = \frac{\vec{V}_{xy}}{\left\|\vec{V}_{xy}\right\|} - \frac{\vec{D}_{xy}}{\left\|\vec{D}_{xy}\right\|} \qquad (9)$$

And a norm of this error can be extracted:

$$E = \left\|\vec{E}_{xy}\right\| \qquad (10)$$

Hence the goal for the brain is to reduce the error E by adapting the set of parameters λ (i.e. adapting b_0, m, and $\overrightarrow{PD}$). One simple way to do this is to perform a gradient descent of the error function with respect to λ. The brain, however, does not have knowledge of the motor transformation performed by the decoder (function PVA in equation 7) is unknown, meaning that the gradient $\frac{\partial E}{\partial \lambda}$ cannot be computed.

Indeed, the error can be expressed as a function of the decoder and target direction:

$$E = \left\|\vec{E}_{xy}\right\| = g(\vec{V}_{xy}, \vec{D}_{xy}) = g(PVA(f), \vec{D}_{xy}) \qquad (11)$$

This implies that learning a set of λ parameters that will allow correct operation of the BMI system must rely on non gradient-based algorithms.

The "learning without gradient" problem is not new, and has been addressed previously in the literature. The basic idea is to get an estimate of the local gradient. For example, [12] proposed to introduce successive parameters perturbations, thus performing a kind of "numerical differentiation" of the error function. This technique is not well suited to problems involving a high number of parameters since it requires a lot of perturbations (and hence of error evaluation) to get an estimate of the gradient. Cauwenberghs [13] introduced an "error descent algorithm" where noise is added to all parameters at a time, thus requiring only one parameters perturbation; the gradient is estimated by computing correlation between the change in parameters and the change in the error. Adding noise to the parameters to estimate the local gradient is a reasonable approach when dealing with inherently noisy processes such as synaptic transmission and neuronal firing rates. Mazzoni et. al. [14] followed the same concept, adapting synaptic weights instead of high-level parameters, thus presenting a more biologically relevant framework. This idea has been recently further developed by [15], where the irregularity of the spiking process was considered the noise source.

C. *Global learning architecture*

During closed-loop BMI, as in classical motor control, two mechanisms will contribute to the production of an appropriate neural command: first an open-loop controller, the inverse model, and second a feedback controller. It has been suggested by Wolpert et. al. [10] that the inverse model is updated based on the feedback that is needed to reduce task error. Thus, the model presented here comprises two modules (see Figure 2): a feedback controller, and an open-loop controller. The feedback controller reduces the error, and based on its actions, the inverse model is updated. This model is detailed in the following subsections.

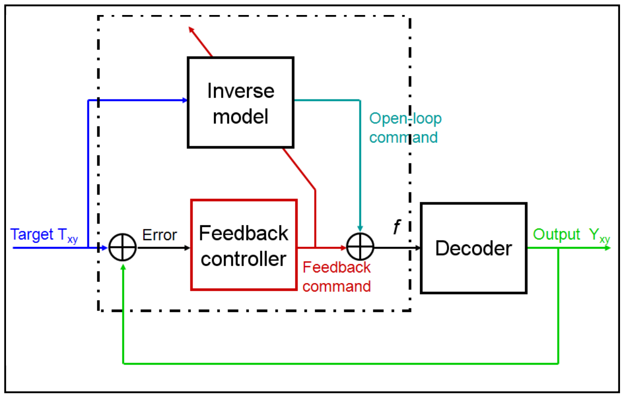

Figure 2. The inverse model is updated based on the corrections made by the feedback controller.

1) *Feedback controller*

We chose to use the “error descent algorithm” proposed by [13] because of its efficiency and biological plausibility. For a given target T_{xy} the open-loop controller generates a neural command $f_{ol} = O(T_{xy}, p)$ that leads to an error E_{ol}. Feedback is added so that the full command $f = f_{ol} + f_{fb}$ leads to smaller error. To achieve this the feedback term is determined using the “error descent algorithm”. First, the reach error E_{fb} is evaluated (through eq.3.) for a given feedback value f_{fb}. Then a random small perturbation p_{fb} is added to the feedback term: $f_{fbp} = f_{fb} + p_{fb}$, and the new reach error E_{fbp} is evaluated. This leads to an error difference term:

$$E_{\Delta} = E_{fbp} - E_{fb} \qquad (12)$$

The following feedback update rule allows statistical descent of the error gradient:

$$\begin{aligned} &\Delta_{fb} = -\mu.E_{\Delta}.p_{fb} \\ &\text{and } f_{fb} = f_{fb} + \Delta_{fb} \end{aligned} \qquad (13)$$

In this scheme, E_{Δ} represents the error contribution due to the perturbation p_{fb}. μ is a strictly positive constant, leading to a statistical decrease of the error. Since numerous cells are involved but a single error measure is available, the update rule (13) does not guarantee that the change in feedback goes in the right direction for a given cell at a given update step. However, the algorithm statistically leads to error descent over the whole population. Hence, it is well suited as a model for the feedback controller: the error is evaluated then reduced by adapting the feedback term f_{fb}.

2) *Open-loop controller*

Let’s assume that the open-loop controller follows the direction tuning neuron model from equation 1. The parameters of individual cells (b_0, m, and $\overrightarrow{PD}$) have to be updated based on the feedback command term (see figure 1). Intuitively, the higher the feedback term is, the more the parameters of the open-loop controller should be changed. The ideal open-loop controller, a perfect inverse model of the decoder, would lead to zero error, hence zero feedback, and to no updates of the open-loop parameters. With a feedback term f_{fb} for cell i, the following update rule can be used:

$$\begin{cases} b_0(t+1) = b_0(t) + \nu.f_{fb}(t) \\ \text{and with } \vec{u} = m(t).\overrightarrow{PD}(t) + m(t).\nu.f_{fb}(t).\dfrac{\vec{D}_{xy}}{\left\|\vec{D}_{xy}\right\|} \\ m(t+1) = \left\|\vec{u}\right\| \\ \overrightarrow{PD}(t+1) = \dfrac{\vec{u}}{m(t+1)} \end{cases} \qquad (14)$$

The parameters are updated so that they incorporate the feedback f_{fb}, which is known to reduce the error. ν is the learning speed, and has to be small. Step by step, the inverse model is built by small increments in the parameter space resulting in a decrease of the cursor movement error $\vec{E}_{xy}$. An additional constraint is added to the parameter b_0 to ensure that the firing rate of the cell remains positive at all time, b_0 is increased if $f<0$.

III. Results

A. Simulations

Closed-loop BMI experiments were simulated using the scheme presented in Figure 1. The PVA algorithm was used to decode neural firing rates. $N=40$ cells were used in these simulations in order to match the average number of cells in experiments performed by [9]. Neurons parameters (b_0, m, and $\vec{PD}$) were assigned randomly, and the PVA parameters were derived accordingly, i.e. the PVA decoder matched the initial tuning of the cells. This led to accurate control of the virtual cursor by the simulated brain. The movement error E was close to 0, thus requiring no changes in the open-loop controller.

Once we checked that the cursor control was stable, the decoding algorithm was perturbed by rotating the tuning function of 10 randomly chosen units by 90° about a common axis, creating a global rotation in decoded cursor movement. This perturbation results in movement errors that require changes in the model parameters.

B. Behavior

Figure 3 (top) shows the evolution of the movement error E across time. As expected, this error decreases over time. This decrement in movement error is well correlated with the time needed to reach the target (bottom). In this simulation, the time allowed to the brain to reach a target was limited to 40 time epochs (corresponding to 4 seconds for firing rates sampled at 10 Hz, which is a typical sampling rate for real-time BMI systems [3]). Initially the system cannot reach the target within this time window. Once the movement error has decreased, the time-to-target also decreases.

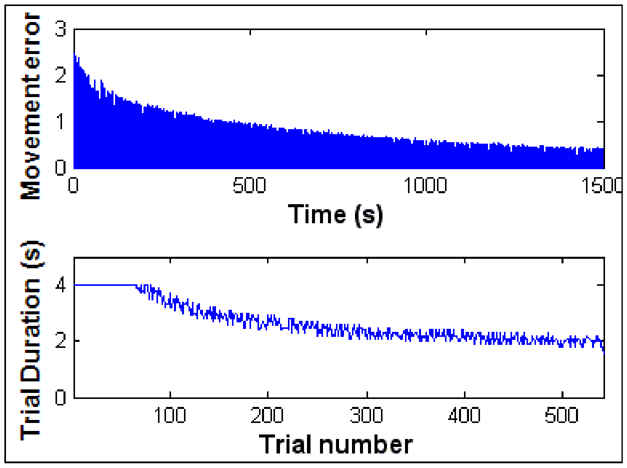

Figure 3. Evolution of movement error (top), and trial duration (bottom).

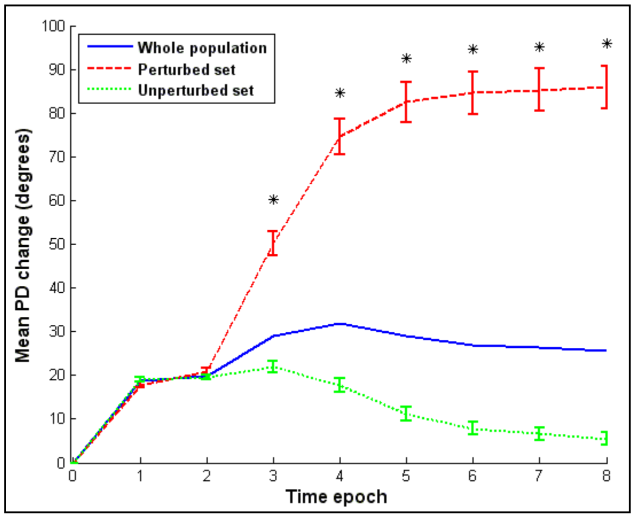

Figure 4. Mean preffered direction changes during simulated closed-loop BMI experiment. Solid blue line reprensents the whole population; red dashed line the set of neurons for which the decoder tuning properties have been rotated; magenta dotted line stands for the set of unperturbed neurons. Errors bars show the standard errors of the mean. * Indicates statistically significant ($p<0.01$) differences between perturbed and unperturbed neurons' PDs.

C. Changes in preferred directions

We analyzed the preferred direction changes that occurred during our simulated learning experiment. As expected, the net direction change of the whole population points in the direction of the perturbation we introduced in the decoder (Figure 4, solid blue line). Two possible strategies that the brain could implement are revealed by the model performance: First, when the model experiences movement errors, its first response is to shift the tuning directions of all neurons to compensate for the perturbation. This is consistent with the "global compensation" observed by [9]. Indeed, early after the perturbation is introduced, there is no statistical difference between the mean change in PDs for neurons for which the PVA tuning function has been rotated compared to unperturbed neurons ("*" in Figure 4 indicate statistically significant differences between the PD changes for perturbed and unperturbed neurons, t test, $p<0.01$). The second strategy arises after this early response: the model identifies neurons for which the decoder tuning function has changed, and adapts their PD accordingly. As a result, the mean PD change for the perturbed set of neurons shifts significantly higher than of the unperturbed set. This is consistent with the larger compensatory tuning changes in the perturbed units than the unperturbed units observed by [9]. Figure 5 shows the distribution of PD changes for the two subsets of cells at two different points in time, corresponding to epoch #1 and epoch #3 in Figure 3. As learning goes on, the preferred directions of the perturbed neurons tend to get closer to the new PVA direction tuning, leading to a change in preferred direction close to 90 degrees. Consequently, the PD changes for the unperturbed set of neurons decrease.

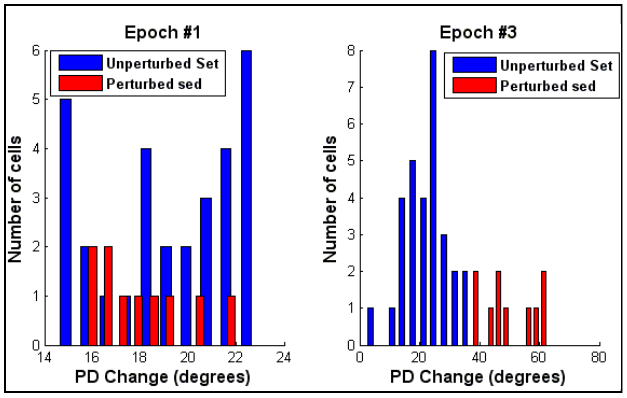

Figure 5. Distribution of PD changes at two different time epochs (corresponding to #1 and #3 in figure 4). In epoch #1, the two distributions look pretty much alike. Later in the learning process, they are statistically different.

IV. Discussion

This works presented a model for motor control learning in a BMI paradigm. Functional reorganization of neurons properties during closed-loop BMI experiments has been reported, and the model presented here is able to simulate those experimental observations. A couple of remarks are worth being made at this point. First, the results presented in section III.C, though not surprising, are not entirely expected from a systems point of view. To respond to the decoder perturbation, the ideal solution for the simulated population of neurons would be to immediately switch the perturbed neurons 90 degrees in the direction of the perturbation, hence matching the perturbed decoder. In average, this is indeed the situation to which the model tends after a long period of learning, as shown in Figure 4. However, before reaching this final state, the system goes through a transition state in which all the neurons, perturbed or unperturbed, experience changes in their preferred direction. We hypothesize that this effect leads to the functional changes experimentally observed by [9]. Second, our simulation enables us to predict the evolution of PD changes over a long period of time. It would be informative to perform closed-loop BMI experiments over a long time scale (i.e. several days) to assess the validity of our model predictions across time. These experiments are however difficult to perform because of the typical instability of the neural recordings.

Modeling the motor learning process during closed-loop BMI has useful applications. First, it can be used to better understand the rules governing the learning process. While the learning rules described in section II.C lead to satisfactory predictions of the behavior and neuronal properties, it could be extended to integrate more experimental observations. By refining these rules, we would identify the important properties of the learning process. In addition, these models will aid in the design of decoders that will maximize the learning speed rather than the offline prediction power of the decoder.

Acknowledgments

This paper was supported in part by the French “Délégation Générale pour l’Armement” under Grant N° PDE 07C0067 to RH, and the Christopher and Dana Reeve Foundation to JMC.

References

[1] Serruya MD, Hatsopoulos NG, Paninski L, Fellows MR, Donoghue JP (2002) Instant neural control of a movement signal. *Nature* 416: 141–142.

[2] Taylor DM, Tillery SI, Schwartz AB (2002) Direct cortical control of 3D neuroprosthetic devices. *Science* 296: 1829–1832.

[3] Carmena, J.M., Lebedev, M.A., Crist, R.E., O’Doherty, J.E., Santucci, D.M., Dimitrov, D., Patil, P.G., Henriquez, C.S., and Nicolelis, M.A.L. “Learning to control brain-machine interface for reaching, grasping by primates,” PloS Biol., vol. 1, pp. 193–208, 2003.

[4] Musallam S, Corneil BD, Greger B, Scherberger H, Andersen RA (2004) Coginitive control signals for neural prosthetics. *Science* 305: 258-262.

[5] Hochberg LR, Serruya MD, Friehs GM, et al. (2006) Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature* 442: 164-171.

[6] Santhanam G, Ryu SI, Yu BM, Afshar A, Shenoy KV (2006) A high-performance brain-computer interface. *Nature* 442: 195-198.

[7] Velliste, M., Perel, S., Spalding, M.C., Whitford, A.S., and Schwartz. A.B. (2008) “Cortical control of a prosthetic arm for self-feeding.” Nature, vol. 453 pp. 1098–1101.

[8] Truccolo, W., Friehs, G.M., Donoghue, J.P. and Hochberg, L.R. (2008) "Primary motor cortex tuning to intended movement kinematics in humans with tetraplegia" J. Neurosci. vol 28(5), pp 1163-1178

[9] Jarosiewicz, B., Chase, S.M., Fraser, G.W., Velliste, M., Kass, R.E. and Schwartz, A.B. (2008) “Functional network reorganization during learning in a brain-computer interface paradigm” PNAS vol. 105(49), pp 19485-19490.

[10] Wolpert, D. Miall, R. and Kawato, M. (1998) “Internal models in the cerebellum,” Trends in Cognitive Science, vol. 2, pp. 338–347.

[11] Georgopoulos AP, Kalaska JF, Caminiti R, Massey JT (1982) On the relations between the direction of two-dimensional ar movements and cell discharge in primate motor cortex. J. Neurosci 2: 1527-1537.

[12] Jabri, M. and Flower, B. (1992)“Weight perturbation: an optimal architecture and learning technique for analog vlsi feedforward and recurrent multilayer networks,” IEEE Transactions on Neural Networks, vol. 3, no. 1, pp. 154–157, 1992.

[13] Cauwenberghs, G. (1992) “A fast stochastic error-descent algorithm for supervised learning and optimization,” in Adv. Neural Information Processing Systems (NIPS*92), vol. vol. 5, pp. pp. 244–251.

[14] Mazzoni, P., Andersen, R. and Jordan M.I. (1991) “A more biologically plausible learning rule for neural networks,” Proc. Natl. Acad. Scie. USA, vol. 88, no. May, pp. 4433–4437.

[15] Xie, X. and Seung, H. S. (2004) “Learning in neural networks by reinforcement of irregular spiking,” Physical Review E, vol. 69, no. 4, pp. 1–10.