Using Neuronal States for Transcribing Cortical Activity into Muscular Effort

Octave Boussaton, Laurent Bougrain

Using Neuronal States for Transcribing Cortical Activity into Muscular Effort*

Octave Boussaton[1] and Laurent Bougrain[2]

***Abstract*— We study the relations between the activity of corticomotoneuronal (CM) cells and the forces exerted by fingers. The activity of CM cells, located in the primary motor cortex is recorded in the thumb and index fingers area of a monkey. The activity of the fingers is recorded as they press two levers. The main idea of this work is to establish and use a *collection of neuronal states*. At any time, the neuronal state is defined by the firing rates of the recorded neurons. We assume that any such neuronal state is related to a typical variation (or absence of variation) in the muscular effort. Our forecasting model uses a linear combination of the firing rates, some synchrony information between spike trains and averaged variations of the positions of the levers.**

I. Introduction

Impressive results exist in the neuroscience literature where the cortical activity of a monkey is used to control a complete robotic arm (see [1], [2]). Here we are concerned with reproducing the movement of two fingers. The recordings used in this study have been taken on a monkey (*macaca nemestrina*) that has been trained to perform a precision grip task. The task of the monkey consisted in clasping two independent levers between its index finger and thumb (Fig. 1) whenever given a visual GO signal.

In these experiments, 33 corticomotoneuronal cells from the hand area of the motor cortex (area 4) were recorded with glass-insulated platinum-iridium microelectrodes (refer to [3] for more details about retrieving and filtering the data). Each experiment is defined by sequences of neuronal activity signals, or *spike trains*, associated to the recorded positions of the fingers. The muscular effort of each finger is seen as a trajectory over time. See Fig. 2 for an example of the muscular activities of both fingers. The muscular effort was sampled at 500 Hz and the neuronal activity at 200 kHz, the latter has been downsampled at 500 Hz.

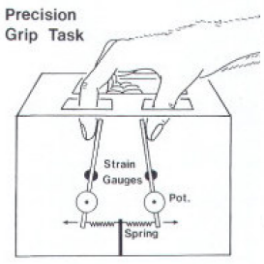

Fig. 1. Experimental setup. The monkey clasps two independent levers hampered by a spring between its index finger and thumb (from [3]).

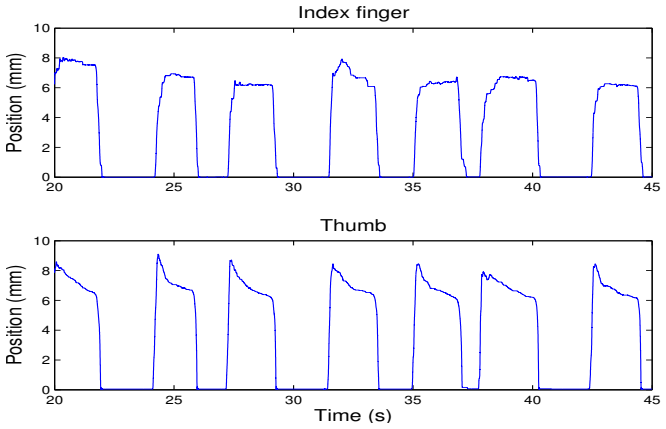

Fig. 2. Coordinated finger movements in millimeters, obtained for the index finger (at the top) and thumb (bottom). This is a 25 seconds excerpt of a 21 minutes recording. Each *bump* is considered as a trial.

According to Hooke's law $F = -kp$, where p is the displacement of the spring's end from its equilibrium position, F is the force exerted by the spring and k is the spring constant. Thus, in the rest of the article, we forecast the position of the fingers as it equals within a constant to the muscular effort. We learn the trajectories of the two fingers separately.

The main objective of this work is to estimate the efficiency of a collection of neuronal states. The method used here is based on a system of first degree linear equations involving the firing rate of the recorded neurons, a set of thresholds associated to them, some synchrony information and the averaged variation of the levers' positions.

II. Method

The muscular effort of each finger is related to the position of the lever it presses. These read as numerical values over the duration of each trial that are considered as trajectories we want to learn. We always consider one finger at a time and write its related trajectory $p(t)$. The prediction at a given time t is noted $\hat{p}(t)$. We suppose $\hat{p}(0) = p(0)$ for coherence (the starting point of the prediction coincides with the observation).

Along with the position of the fingers, each experiment is also defined by spike train signals written $n(t) = (n_1(t), n_2(t), ..., n_N(t))$ where N is the number

*This work was in part supported by CNRS-PIR-Neuro-IC 2010 (Project:CorticoRobot).

[1] O. Boussaton is with Cortex project team, Inria Nancy-Grand Est, Villers-les-Nancy, France `octave.boussaton at inria.fr`

[2] L. Bougrain is with the University of Lorraine and the Department of Complex Systems & Artificial Intelligence, LORIA, Vandoeuvre-les-Nancy, France `bougrain at loria.fr`

of recorded neurons in the experiment. At each time t, $n_i(t) = 1$ if an action potential has been detected at time t for neuron $i \in [1, N]$ and 0 otherwise. Let us write T the length of a given trial.

A. Firing rate functions

First, we compute the **firing rate function** of every neuron i according to a time-window of length w. Simple rate coding is commonly admitted not to be enough for representing all the information contained in spike trains [4]. So we introduce some decay in the propagation of the neuronal information. This is often referred to as using a *spike density function*. We write it here, $\forall i \in [1, N]$:

$$\forall t \in [w-1, T],\ d_i(t)_w = \sum_{k=t-w+1}^{t} n_i(k) \times decay(k,t)_w \quad (1)$$

where $decay(k,t)_w$ is a monotonically increasing function defined on an interval of length w such that $\forall k \in [t-w+1, t]$, $decay(k,t)_w \in [0,1]$ and $decay(t,t)_w = 1$.

We experimented several ways for considering some decay in the neuronal information, a range of linear and exponential functions, logarithmic functions, etc. For example, if we consider of a **linear decay** function, the general form of $decay(k,t)_w$ in (1) would be:

$$decay(k,t)_{w,v} = (1-v) \times \frac{w-t+k}{w} + v \quad (2)$$

where v is the value of the earliest spike in the time window, $v \in [0,1]$, note that $v = 1$ means no decay is considered.

In the case of a squared **exponential decay**, $decay(k,t)_w$ in (1) could be such that:

$$decay(k,t)_{w,v} = (1-v) \times \frac{(e^{\frac{w-t+k}{w}} - 1)^2}{(e-1)^2} + v \quad (3)$$

The linear and even more the squared exponential decay reduce the size of the collection of neuronal states (see section II-B).

In what follows, we omit the window length w and the value v of the earliest spike in the notations as they stay the same after being chosen once.

B. The space of Neuronal States

We call $d(t) = (d_1(t), d_2(t), ..., d_N(t))$ the **neuronal state** or *neuronal code* of the monkey at time t. Depending on the length of the time-window chosen to compute the firing rate functions, each firing rate function d_i has a maximum value max_i. The vector space of all possible neuronal states is *contained* within $D = \prod_{i=1}^{N}[0, max_i]$. For any experiment, at any time t, the neuronal state $d(t)$ of the monkey belongs to D.

Our initial assumption is that a given neuronal state implies a specific variation in the muscular effort (or an absence of variation). Here this means a certain variation of the position and so its derivative.

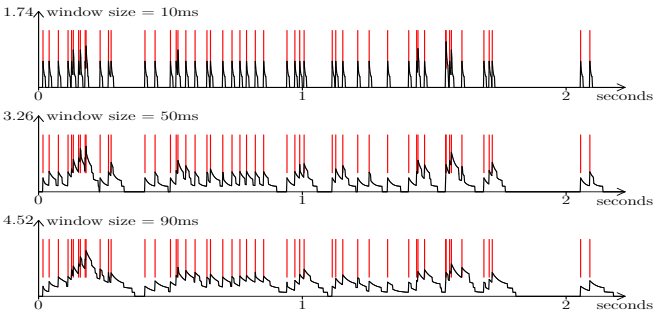

Fig. 3. Firing rate functions of the same spike train computed with a squared exponential decay for different sizes of the time window. Vertical lines indicate spikes. Maximal values of the firing rate functions are also reported on the left in the picture.

Remark: In order to establish a collection of neuronal codes, we assume that D is a discrete space and round the firing rate values of any encountered neuronal code, which is a vector, to the closest integer valued vector, i.e. $[1.23, 0.47, 5.63, 6.15]$ would be referred to as the *discretized* neuronal code $[1, 0, 6, 6]$. It is easy to see that a granularity of 0.5 roughly increases the size of the neuronal code collection by a factor of 2^N and thus the computation time, for this reason we kept using a granularity of 1.

Fig. 3 shows the influence of the window size on the firing rate functions and the maximal values of the firing rates. These have a direct influence on the size of the collection of neuronal codes, given the definition of D, so they also influence the computation time the learning process requires: the smaller the faster.

These functions are not continuous, for example [4] contains ways for solving this issue but we do not need these functions to be continuous in our computation.

1) Different sizes for the collection of neuronal codes: The size of the time window used for computing the firing rate functions affects the size of the collection of all neuronal codes present in the learning set, the more codes in it, the more parameters subject to optimization.

We tested different sizes for the time windows in the range 10 to 100 milliseconds and different functionals for the decay (see Fig. 4). The evolution in size of the collection has almost the same shape for the same sizes of learning sets for any type of decay.

2) Three phases for the muscular activity: Using the neuronal states seems to be a good idea but the problem when not enough neurons are recorded is that *key* neuronal codes appear many times with many very different values for the derivative of the related finger position. This reduces the global accuracy of the whole learning. In order to counterbalance this phenomenon, we defined three phases for the muscular activity and thus for the neuronal states. These can be either *decrescent, steady or crescent*. For the trials in the learning set, this is done according to the recorded muscular activity, for the trials in the estimation set, according to what has been learned. In practice, to determine the phase at a given time t, we

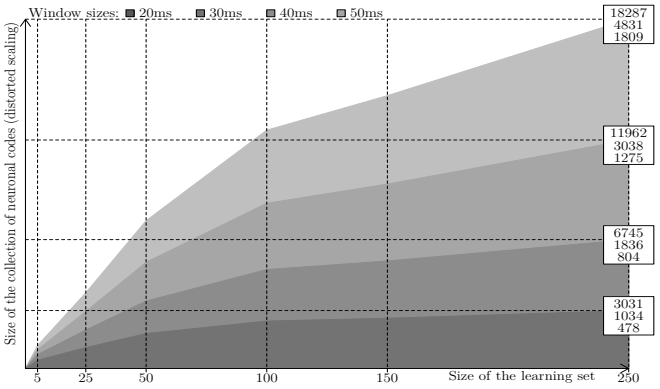

Fig. 4. Different sizes for the collections of neuronal codes for respectively, no decay, linear decay, exponential decay (the numbers on the right part of the picture) when the firing rate functions are computed with different sizes of learning sets. 6 neurons were recorded in these experiments.

averaged the derivative on the 10 milliseconds prior to instant t, if greater than 0.01 it is considered crescent, smaller than -0.01 is decrescent, values in between are considered steady.

C. Synchrony information

Moreover, we compute synchrony information between each possible pair of neurons that is propagated for a period of time via the firing rate of **synchrony trains**.

Let $n_{i,j}$ be the synchrony train of spike trains n_i and n_j and let w_s be a synchronicity period. Then $n_{i,j}(t) = 1$ if both neurons i and j emitted *at least* one action potential between times $t - w_s + 1$ and t, $n_{i,j}(t) = 0$ otherwise.

The firing rate functions of these synchrony trains are computed in the exact same way as the firing rate functions of the *original* spike trains. If N neurons are recorded, $C_2^N = N(N-1)/2$ synchrony trains are computed according to a time window of length w_s. The synchrony train of trains i and j, written $d_{i,j}$, is such that:

$$d_{i,j}(t) = \sum_{k=t-w+1}^{t} n_{i,j}(t) \times decay(k,t)_w.$$

D. Average derivative values

Every neuronal code present at least once in the learning set is associated to the average variation, or derivative value, of the observed positions for this code. Let us write $\Delta(p(t)) = p(t) - p(t-1)$ the variation in the finger's position for trial p at time t. We compute the average variation $\bar{\Delta}(d)$ of each unique neuronal state $d \in D$ observed in the learning set. For any neuronal state d that does not appear in the learning set, we suppose $\bar{\Delta}(d) = 0$.

III. The learning process

We start by setting up the learning set, on which the forecasting formula is optimized (see (4)), and the estimation set, on which the quality of the learning is evaluated. Different sessions have been recorded involving from 3 to 6 neurons. Each session includes around 300 trials. Each trial contains between 1500 and 2500 samples. Usually, 60% of the total length of a session is used for training and 40% for testing.

First, we identify all the different neuronal states encountered in the learning set and associate them with the average derivative value of the related muscular effort (see II-D). Then we aim at optimizing the following forecasting formula:

$$\hat{p}(t+1) = \hat{p}(t) \; + \; A(t) \times \bar{\Delta}(d(t)) \tag{4}$$

where $A(.)$ is a ponderation of the average derivative value at time t. It is based on linear combinations of the firing rate functions and, for each one of them, one ponderation thresholds and two ponderation coefficients.

A. Ponderation of the derivative

In case $d(t)$ in the above formula does exist, we have:

$$\begin{array}{rcl} A(t) & = & \sum_{i=1}^{N} d_i(t) \cdot H_i(d_i(t), \theta_i) + \\ & & \sum_{i,j \neq i} d_{(i,j)}(t) \cdot H_{(i,j)}(d_{i,j}(t), \theta_{(i,j)}) \end{array} \tag{5}$$

where θ_* are thresholds and $H_*(a,b)$ is a bi-valued functional such that

$$\begin{array}{rcl} H_*(a,b) & = & h_{*,1} \text{ if } a \text{ is smaller than } b. \\ & & h_{*,2} \text{ otherwise.} \end{array}$$

Ponderation thresholds θ_* and coefficients $h_{*,1}$ and $h_{*,2}$ are the parameters that vary as the learning goes on. We have $N + C_2^N$ thresholds and twice that number of coefficients. Then the size of the collection of the neuronal codes is a fixed number depending on the learning set.

B. Learning parameters

Our learning process optimizes successively, one cycle after another, each set of parameters : the ponderation coefficients, the ponderation thresholds and the averaged movement variations recorded in the collection of neuronal codes. In a learning cycle, all the parameters of a certain type get modified individually, if the modification reduces the global error on the learning set, it is considered as an improvement and kept. It is undone otherwise.

Finally, during the optimization cycle that is concerned with the average variations observed in the collection of neuronal codes, we use the aforementioned phases to specify these modifications. Modifications to steady states are done with a factor 0.001 compared to others 0.01.

Initially, all ponderation thresholds are set to random values between 0 and half the max firing rate of the spike train they are related to. The initial values of the ponderation coefficients $h_{*,1}$ are close to 0, regardless of any scaling factors, and we start with $h_{*,1} = h_{*,2}$.

IV. Results

We tested different settings, in order to find out the best decay method, the optimal window size, propagation period etc. These comparisons can be partially summed up in Table 1 in which are reported various values for the mean absolute error obtained for the estimation set.

TABLE I

MEAN ABSOLUTE ERROR (MAE) / SIZE OF THE COLLECTION OF NEURONAL CODES IN DIFFERENT CASES. 65 TRIALS WERE USED FOR LEARNING AND 15 FOR ESTIMATING. FOR THE LINEAR AND SQUARED EXPONENTIAL DECAYS, THE VALUE IN PARENTHESES IN THE FIRST ROW OF THE TABLE IS THE VALUE OF v.

Size	No decay	Exp.(0.1)	Exp.(0.4)	Lin.(0.1)	Lin.(0.4)
20ms	1.28/*317*	1.35/*85*	1.11/*139*	1.17/*154*	1.16/*192*
30ms	1.21/*541*	1.19/*135*	0.79/*225*	0.92/*223*	1.12/*312*
40ms	1.75/*779*	0.85/*165*	0.74/*306*	0.85/*316*	1.06/*453*
50ms	1.26/*1056*	0.97/*211*	0.74/*403*	0.83/*415*	1.13/*583*
60ms	1.47/*1343*	0.83/*251*	0.94/*481*	1.10/*495*	1.26/*746*
70ms	2.11/*1647*	0.84/ *296*	0.82/*589*	1.24/*621*	1.11/*911*

The simplest case where no decay is considered is the least accurate. A linear decay allows to reduce the size of the collection of neuronal codes. The exponential case reduces it even more, along with improving the results in most cases.

The best results were obtained with:

- an exponential decay where the earliest spike of the time window counts for 0.4;
- a time window of length 30 to 40 ms for computing the firing rate functions;
- a synchronicity period of length between 20 to 30 ms;
- a learning set of size between 50% and 75% of the total set considered.

Fig. 5 shows the quality of our method on a difficult case. Indeed, here there is a *gash* at the 3 seconds mark in the monkey's activity but still, the forecasting formula identifies it.

V. DISCUSSION & CONCLUSION

The collection of neuronal states introduced in this paper is efficient for transcribing the cortical information into muscular effort. Whereas artificial neural network models often count neural discharges in 100 ms bins ([5], [1]), our linear model relies on time windows for establishing the firing rate functions is optimal between 20 and 30 ms, depending on the decay function being used. Our learning method can be used in real time, once a collection of neuronal code has been established, since the firing rates and synchrony information can be easily actuated at each time step.

The decay functions allow to drastically reduce the size of the collection of neuronal codes, as shown in Fig. 4 where in the case of recordings with six neurons, the size of the collection can be lowered by about 78% between the "no decay" and the "exponential" cases. We showed that they also enhance the results. According to our experiments, a squared exponential decay function as we defined gives the most satisfying results but others are not too far behind.

It is interesting to note that longer time windows for computing the firing rate functions do not yield better performance with our method. In fact it is the opposite: down to 20 ms, the smaller the time window the better

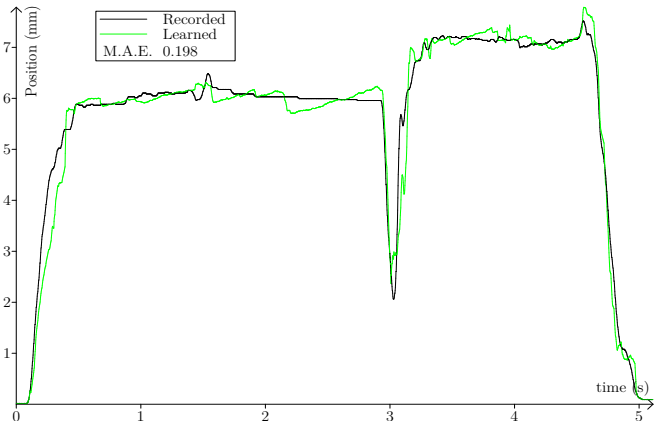

Fig. 5. An estimation after two cycles of improvement on the ponderation coefficients and thresholds, then an improvement on the collection of neuronal codes. The time window was 20 ms here, for a squared exponential decay. The learning was done on 40 trials, evaluated on 10 *hard* ones and this is the best result.

the results. The optimal length for the time windows that are used for computing the firing rate functions is consistent with [6].

Considering different phases for better identifying the muscular effort is obviously also useful, this makes the learning much more efficient on the slope parts of the curve.

With this reasonably simple method, we obtain average learning errors smaller than 1, which means that in average, the predicted position of the finger is off by maximum 1 millimeter.

ACKNOWLEDGMENT

We are very thankful to R. Lemon and M. Maier for providing the data set to us and to Thierry Vieville for his precious collaboration.

REFERENCES

[1] J. M. Carmena, M. A. Lebedev, R. E. Crist, J.E. O'Doherty, D. M. Santucci, D. F. Dimitrov, P. G. Patil, C. S. Henriquez, and M. A. L. Nicolelis, "Learning to control a brain machine interface for reaching and grasping by primates," *PLoS Biology*, vol. 1, no. 2, pp. 193–208, 2003.

[2] M. Velliste, S. Perel, S. Chance, A. Spalding, and A. Schwartz, "Cortical control of a prosthetic arm for self feeding," *Nature*, vol. 453, pp. 1098–1101, 2008.

[3] M. Maier, K. Bennett, M. Hepp-Raymond, and R. Lemon, "Contribution of the monkey corticomotoneuronal system to the control of force in precision grip," *Journal of neurophysiology*, vol. 18(3), pp. 772–785, 1993.

[4] Attila Szücs, "Application of the spike density function in analysis of neuronal firing patterns," *Journal of neuroscience methods*, vol. 81, pp. 159–167, 1998.

[5] J. Wessberg, C. R. Stambaugh, J. D. Kralik, P. D. Beck, M. Laubach, J. K. Chapin, J. Kim, S. J. Biggs, M A Srinivasan, and M. A. L. Nicolelis, "Real-time prediction of hand trajectory by ensembles of cortical neurons in primates," *Nature*, vol. 408, pp. 361–365, 2000.

[6] Maik C. Stuttgen and Cornelius Schwarz, "Psychophysical and neurometric detection performance under stimulus uncertainty," *Nature Neuroscience*, vol. 11, no. 9, pp. 1091–1099, 2008.