34th Annual International Conference of the IEEE EMBS
San Diego, California USA, 28 August - 1 September, 2012

Brain-Machine Interface Control of a Robot Arm using Actor-Critic Reinforcement Learning*

Eric A. Pohlmeyer, Babak Mahmoudi, Shijia Geng, *Student Member, IEEE*, Noeline Prins *Student Member, IEEE*, and Justin C. Sanchez, *Member, IEEE*

Abstract— **Here we demonstrate how a marmoset monkey can use a reinforcement learning (RL) Brain-Machine Interface (BMI) to effectively control the movements of a robot arm for a reaching task. In this work, an actor-critic RL algorithm used neural ensemble activity in the monkey's motor cortex to control the robot movements during a two-target decision task. This novel approach to decoding offers unique advantages for BMI control applications. Compared to supervised learning decoding methods, the actor-critic RL algorithm does not require an explicit set of training data to create a static control model, but rather it incrementally adapts the model parameters according to its current performance, in this case requiring only a very basic feedback signal. We show how this algorithm achieved high performance when mapping the monkey's neural states (94%) to robot actions, and only needed to experience a few trials before obtaining accurate real-time control of the robot arm. Since RL methods responsively adapt and adjust their parameters, they can provide a method to create BMIs that are robust against perturbations caused by changes in either the neural input space or the output actions they generate under different task requirements or goals.**

I. Introduction

Brain-Machine Interfaces (BMIs) are computer systems that allow a user to control a device's actions directly from their neural activity. Such systems have shown great potential to restore motor function to people living with amputation and paralysis. Numerous tests in animal models, as well as some human tests, have already shown the feasibility of using BMIs to control a wide range of devices including: communication systems, computer cursors, robot arms, and functional electrical stimulation systems [1-8].

The push to move these demonstrations beyond proof-of-concept is motivated by the need to use them during activities of daily living to provide greater independence for persons living with paralysis or amputation. Such a change in deployment introduces new challenges to BMI design. To date, most BMI tests share a reliance on supervised learning methods (Wiener filters, neural networks, generative models, etc.) to create the BMI control module. In supervised learning, an explicit set of training data is used to directly map the system inputs onto the outputs to create the model, which than remains constant unless explicitly updated using a modified set of training data. For example, BMI tests that involve healthy nonhuman primates often involve a training phase in which the animal achieves the goal of the BMI task directly with actual arm movements [2, 6-8], this data is then used to relate the task outputs (e.g. cursor or robotic movements) to the neural modulation. Even in cases where real arm movements are not explicitly used during the training, a paradigm must be employed in which the desired output from the BMI is carefully controlled so it can be related in some way to the inputs [1, 4]. After the training mode, the algorithm is then 'frozen', and then used to allow neural modulations to drive the task directly in a true BMI mode. Since the BMI algorithm parameters have been fixed, changes for improving or modifying the performance of the system are only accomplished by the brain adapting its neural activity to better work with the computer [6]. By using such fixed control algorithms, the BMIs become sensitive to perturbations in either the inputs (e.g. neurons appearing or disappearing from the neural recordings), new environments, or in changes in the output (e.g. cursor movements unrepresented in the training data becoming desired). This can limit the long-term efficacy of the BMI, and requires the BMI to be regularly updated through additional training sessions, which reduces user independence.

Reinforcement learning (RL) is a method of machine learning that is modeled after biological mechanisms and which does not rely on training data. In RL, the system adapts itself through experience and its interactions with the environment [9]. Since BMI's based on RL [10, 11] can 'start from scratch' with no prior knowledge of the desired inputs and outputs, and then iteratively refine the model parameters, they would be a natural fit for real BMI applications in which the user was unable to manually drive the system to provide training data. Furthermore, since in RL the model weights can be continually refined and updated, such algorithms may be more effective than supervised methods in providing BMIs that are robust to changes in the input, output or environmental spaces.

II. Methodology

A. Neural Recordings

Neural data was acquired from the motor cortex and nucleus accumbens (NAcc) of a single marmoset (*Callithrix jacchus*) monkey. Multielectrode arrays (16-channel tungsten microelectrode arrays, Tucker Davis Technologies, FL) were surgically implanted in each brain region under isoflurane anesthesia and sterile conditions. Neural data was acquired using a Tucker Davis Technologies RZ2 system at 24,414Hz. Neuronal signals were discriminated in real-time based on the waveform amplitudes and shapes and using manually set threshold levels. Both multiunit signals as well as well-isolated single unit signals (collectively referred to here as neuronal signals) were recorded and used equivalently in all

*This work was supported by DARPA REPAIR project N66001-10-C-2008.

E. A. Pohlmeyer, B. Mahmoudi, S. Geng, N. Prins, and J. C. Sanchez are with the Department of Biomedical Engineering, Miami University, Coral Gables, Fl 33146 USA (e-mail: jcsanchez@miami.edu).

applications. During the real-time experiment, 21 motor and 18 NAcc signals were recorded. All surgical and animal care procedures were consistent with the National Research Council Guide for the Care and Use of Laboratory Animals and were approved by the University of Miami Institutional Animal Care and Use Committee.

B. Experimental Task

The monkey was trained to move a robot arm to one of two spatial targets to receive food rewards, as shown in Fig. 1. To create contrast in the value of each target, the monkey was conditioned to associate one target with a desirable food treat (waxworm or marshmallow, 'A' trials), and the other target with an undesirable object (wooden bead, 'B' trials). The monkey began a trial by placing its hand on a touchpad for a random hold period (0.7-1.2 seconds), an audio go signal was then administered that corresponded to a robot arm moving upwards, out from behind an opaque shield, and presenting its gripper (holding either the desirable or undesirable object) to the monkey. Simultaneously, the A or B LED spatial target corresponding to the type of object in the gripper was illuminated. When in manual control mode, the monkey moved the robot to the A target by reaching to a second sensor (2 second reach time limit), and to move the robot to the B target the monkey was required to keep its hand motionless on the touchpad for 2.5 seconds. Thus, the monkey was taught to associate changes in motor activity during A trials compared to static motor responses during B trials. The monkey was given food rewards (waxworms/marshmallows) after having moved the robot to the illuminated target for both the A and B trials equivalently.

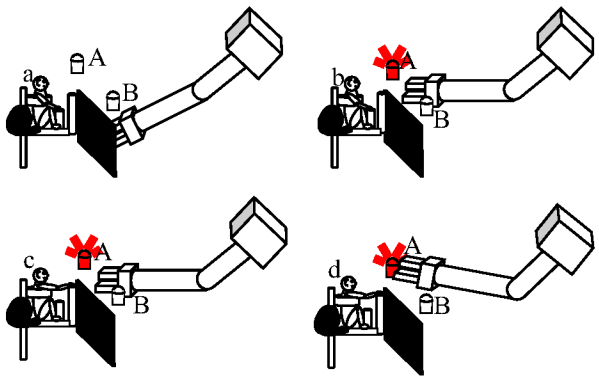

Figure 1. The monkey was required to move a robot arm to one of two LED targets to receive a food reward. a) monkey triggers trial b) Robot moves out from opaque screen, A target LED lights c) Monkey makes arm movement d) Robot moves to A target

C. Actor Critic RL BMI Robot Arm Controller

In BMI control mode, the robot movements were determined directly from the monkeys' motor cortex activity in real-time using an actor-critic RL algorithm [9]. As shown in Fig. 2, the input to the RL algorithm was the state of the motor cortex (vector of firing rates of motor neuronal signals during a 2 second window following the trial go tone), and the outputs were "values" associated with both possible robot actions (movements). The goal of an RL actor is to find the value of state-action pairs [9]. The robot then chose the action (A or B) that had the higher value, i.e. a greedy "policy" in RL. The actor used here was a fully connected, 3-layer, feedforward neural network combined with a Hebbian structure [12, 13]. The first layer received the neural firing rates and the hidden layer contained five binary stochastic elements. These neurally inspired computing elements performed a weighted sum of synaptic inputs and computed a probability of firing (range of -1 to 1) using a hyperbolic tangent function. Once the signals passed through these basis functions, they were then mapped to the output action values.

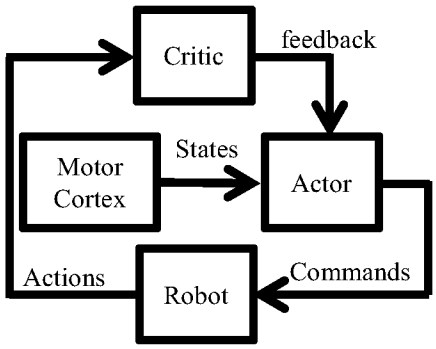

Figure 2. Actor-critic reinforcement learning control architecture. Feedback from the Critic is used to improve the Actor's control the robot using the monkey's neural states.

The adaptation of model weights used by RL methods differs from supervised learning decoding techniques. The actor critic RL method used here is a semi-supervised machine learning method in which a neural decoder (actor) learns via trial and error experience with its environment. Weights are initialized randomly and then iteratively updated as each new set of data (trial) is experienced by the algorithm. The changes to the weights are based on an evaluative feedback signal, provided by the critic module after an action selection, which assesses the efficacy of the previous selection. In these experiments, if the previous action selection was correct the feedback was +1 and -1 otherwise. In simple terms, the decoding algorithm was provided with a basic performance score of how well it completed the task, and over time the weights were iteratively updated as each new set of data was experienced by the algorithm. Mathematically, all the weight updates were computed as:

$$\Delta \boldsymbol{w_{ij}} = \gamma f\left(\boldsymbol{x_i}(\boldsymbol{p_j} - \boldsymbol{x_j})\right) + \gamma(1-f)\left(\boldsymbol{x_i}(\mathbf{1} - \boldsymbol{p_j} - \boldsymbol{x_j})\right) \quad (1)$$

Here w_{ij} is the synaptic weight connecting nodes i and j, γ is the learning rate, p_j is a sign function of tanh output x_j (positive values become +1 and negative values become -1) and f is feedback from the critic. Notice that the update consists of the multiplication of the feedback and the occurrence of firing (via the sign function p) between node inputs and outputs (x_i and x_j, respectively). Thus, if nodes of the actor-critic network fire together their synaptic weights are adjusted proportionally to the feedback signal. This is the essence of Hebbian style learning, although here it is implemented in a reinforcement learning framework.

In order to speed the initial adaptation from the purely random initialization weights to functionally useful weights during closed-loop brain control, real time 'epoching' of the data was also used. Rather than the weights being updated simply with the single most recent trial's data, saved data from all previous trials were rerun through the RL update

equations after each trial (ten times following each new trial). Finally, the neuronal firing rates were normalized (-1 to 1) before being used as inputs. This was accomplished by keeping a real time record of the highest firing rate detected for each input, which was used to continually update the normalization parameters throughout the session.

III. Results

A. Electrophysiology

The monkey quickly learned to control the robot arm. In sessions where the monkey controlled the robot manually, the monkey typically performed 50-60 trials (mean accuracy of 90% for the first 50 trials, 8 manual sessions) before beginning to lose motivation. The task was quite effective in teaching the monkey consistently evoke different motor states. Fig. 3 shows an example histogram of the neuronal signals from one electrode when the monkey was controlling the robot arm either manually (top panel) or via the BMI (bottom panel). Time zero corresponds to the trial go cue. In A trials an upregulation in firing is obvious after the cue, while in B trials firing levels were maintained.

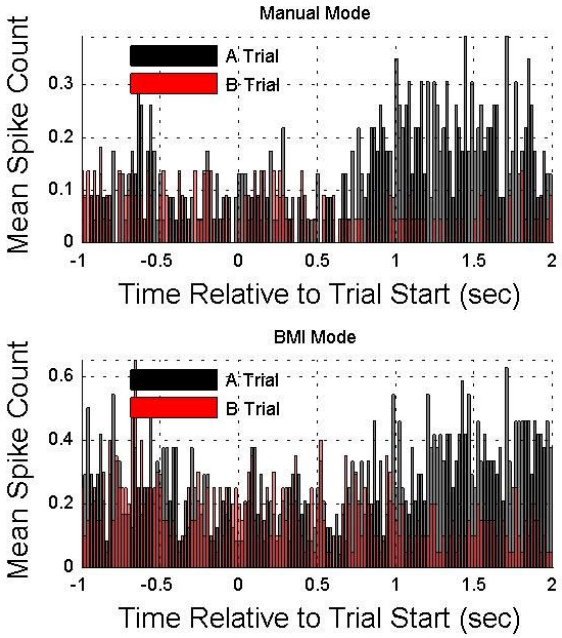

Figure 3. The monkey associated two basic motor states with the behavioral task. Histograms from a single electrode when the monkey was manually controlling the robot arm (top) and when the arm was driven by the BMI (bottom) show the similarity in response under both conditions. Black is response during A trials, red during B trials.

B. Reinforcement Learning BMI Performance

Fig. 4 shows the monkey's performance during a single session in which it used the actor-critic RL BMI to control the robot arm. The model weights were initialized randomly from a uniform distribution (-0.075 to 0.075). For a brief initial period, the monkey was unable to effectively control the arm, but the RL algorithm was able to converge to consecutive correct responses within 5 trials, and remained effective until the monkey was no longer interested in the task. Overall throughout the session the robot moved correctly to the cued target in 85% of the trials. Furthermore, in most cases where the robot moved to the incorrect target, this action was attributed to the monkey producing the incorrect motor state. For example, during almost all BMI A trials the monkey still made arm movements (despite those not being directly necessary to control the robot), this included trials 1 and 3. The incorrect robot movements during those trials thus indicated the RL algorithm was still learning an effective set of weight parameters. Conversely, in A trials 18, 27, 42, and 43 (highlighted in Fig. 4), the monkey did not move his arm and the RL algorithm moved the robot to the B (nonmove) target. Similarly, in trial 23 (B trial), while the monkey did not make an overt reaching movement he did move around in the chair, and the RL BMI moved the robot to the A target. This suggests that the model correctly interpreted the monkey's motor neural state in 94% of the trials, and most incorrect robot movements resulted from the monkey generating the wrong command signal during lapses in its interest with the task.

Fig. 5 shows the iterative adaptation of the output node weights throughout the task. The algorithm continually changed the weights, although the values begin to plateau after sufficient trials of high performance, with perturbations occurring during trials where the monkey did not correctly participate in the task (e.g. trials 23, 27), which caused the model to readjust itself.

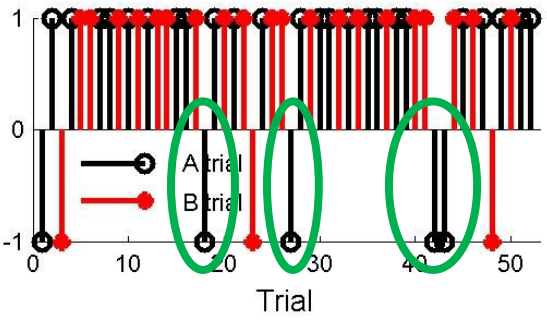

Figure 4. Real time BMI performance. +1 indicates the robot moved to the correct target. The RL algorithm began effectively moving the robot to the correct target after only a few trials. Highlighted trials indicate erroneous robot movements that resulted from the monkey not participating in the task correctly.

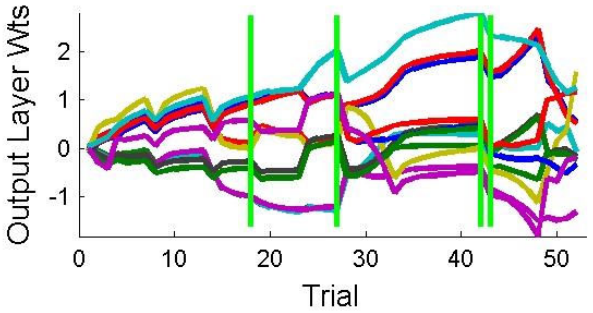

Figure 5. Output layer weight tracks (12 tracks: 5 hidden nodes, 2 output nodes, and 2 offset terms) for the same data pictured in Fig. 4 (vertical lines show Fig. 4's highlighted trials). The adaptation of the algorithm output weights from random initial conditions over time is evident. The adaptation of the hidden layer weights (not shown) followed a very similar pattern.

IV. Discussion

This robot control task demonstrates how reinforcement learning methods can be used as a BMI controller. The algorithm was able to quickly (less than 5 trials) and effectively modify its weights to allow a marmoset monkey to use the BMI to control a robotic arm. Furthermore, the algorithm continued to update the weights as new data was acquired. These updates simply refined the weights without hurting performance so long as the input space remained consistent. The refinement did not cause the RL method to become inflexible to new data. Trials in which the monkey did not attempt to generate the correct control signal caused immediate changes in the weight values, although with little effect on the overall (already good) performance. This is promising as it suggests the algorithm can rapidly respond to changes to the input space and quickly make updates, but can treat such deviations as outliers if doing so maintains overall performance. Intrinsic in this approach is a balance between computational learning and generalization.

A limited amount of epoching of previous trial data was used in this experiment to speed the initial acquisition of effective weights. For BMIs, this is an underutilized approach that can expedite the acquisition of control solutions. In addition, it has a biological analogue of replay that is used during sleep. From a practical perspective though, it may be undesirable to indefinitely store and replay previous data, and sliding windows may need to be used during long term use. These windows may also affect the algorithm's ability to rapidly respond to changes in the input space, or to significantly change action values related to a previously useful action that was no longer desired (or conversely change the weights related to an action that previously been used only sporadically but had suddenly become more useful). Thus, deeper investigation of epoching could yield insight into the balance of how previous and recent data affects updates and action selections.

Finally, the only data needed to continually train and update the algorithm (total of 122 weight parameters) was a simple binary "good/bad" assessment of the effectiveness of the previous decision. While this information was input externally by the computer during this experiment, it is possible that it could be obtained directly from the brain itself. For example, Fig. 6 shows how during the manual control experiments a NAcc neuronal signal showed different modulation during A trials when the robot moved to the target instructed by the monkey, compared to 'catch' trials in which the robot was suddenly moved to the B target despite the monkey having generated the correct A command (and thus was presumably disappointed by the robot moving to the unwanted wrong target). Such information could be used to provide the error feedback for the RL algorithm directly from the brain. RL methods can use such simple training signals to refine algorithms that in turn implement far more complicated tasks that involve many more degrees of freedom than the simple binary action choice employed here [14]. Thus, extracting such simple training data from the brain may enable the creation of adaptive BMI systems that continually refine themselves and thus remain robust against the perturbations and changes in the input space that can cause supervised learning algorithms to lose functionality.

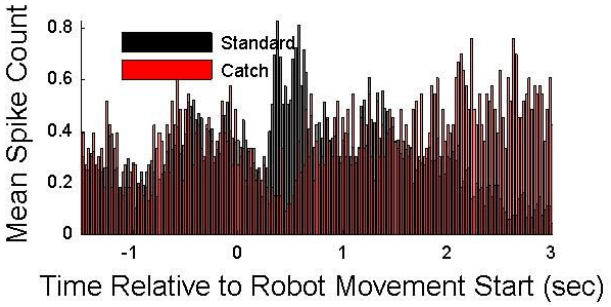

Figure 6. Example NAcc signal during A trials when the robot moved to the wrong target. In random catch trials the robot moved to the B target despite the monkey having generated the correct A command, resulting in no reward.

References

[1] M. Velliste, S. Perel, M. C. Spalding, A. S. Whitford, and A. B. Schwartz, "Cortical control of a prosthetic arm for self-feeding," *Nature,* vol. 453, pp. 1098-101, Jun 19 2008.

[2] E. A. Pohlmeyer, E. R. Oby, E. J. Perreault, S. A. Solla, K. L. Kilgore, R. F. Kirsch, and L. E. Miller, "Toward the restoration of hand use to a paralyzed monkey: brain-controlled functional electrical stimulation of forearm muscles," *PLoS ONE,* vol. 4, p. e5924, 2009.

[3] J. R. Wolpaw and D. J. McFarland, "Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans," *Proc Natl Acad Sci U S A,* vol. 101, pp. 17849-54, Dec 21 2004.

[4] L. R. Hochberg, M. D. Serruya, G. M. Friehs, J. A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, and J. P. Donoghue, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature,* vol. 442, pp. 164-71, Jul 13 2006.

[5] L. A. Farwell and E. Donchin, "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials," *Electroencephalogr Clin Neurophysiol,* vol. 70, pp. 510-23, Dec 1988.

[6] J. M. Carmena, M. A. Lebedev, R. E. Crist, J. E. O'Doherty, D. M. Santucci, D. Dimitrov, P. G. Patil, C. S. Henriquez, and M. A. Nicolelis, "Learning to Control a Brain-Machine Interface for Reaching and Grasping by Primates," *PLoS Biol,* vol. 1, pp. 193-208, Nov 2003.

[7] G. Santhanam, S. I. Ryu, B. M. Yu, A. Afshar, and K. V. Shenoy, "A high-performance brain-computer interface," *Nature,* vol. 442, pp. 195-8, Jul 13 2006.

[8] M. D. Serruya, N. G. Hatsopoulos, L. Paninski, M. R. Fellows, and J. P. Donoghue, "Instant neural control of a movement signal," *Nature,* vol. 416, pp. 141-2, Mar 14 2002.

[9] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, Mass.: MIT Press,, 1998.

[10] J. DiGiovanna, B. Mahmoudi, J. Fortes, J. C. Principe, and J. C. Sanchez, "Coadaptive brain-machine interface via reinforcement learning," *IEEE Trans Biomed Eng,* vol. 56, pp. 54-64, Jan 2009.

[11] B. Mahmoudi and J. C. Sanchez, "A symbiotic brain-machine interface through value-based decision making," *PLoS ONE,* vol. 6, p. e14760, 2011.

[12] P. Mazzoni, R. A. Andersen, and M. I. Jordan, "A more biologically plausible learning rule than backpropagation applied to a network model of cortical area 7a," *Cereb Cortex,* vol. 1, pp. 293-307, Jul-Aug 1991.

[13] P. Mazzoni, R. A. Andersen, and M. I. Jordan, "A more biologically plausible learning rule for neural networks," *Proc Natl Acad Sci U S A,* vol. 88, pp. 4433-7, May 15 1991.

[14] P. M. Pilarski, M. R. Dawson, T. Degris, F. Fahimi, J. P. Carey, and R. S. Sutton, "Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning," *IEEE Int Conf Rehabil Robot,* vol. 2011, pp. 1-7, Jun 2011.