2021 43rd Annual International Conference of the
IEEE Engineering in Medicine & Biology Society (EMBC)
Oct 31 - Nov 4, 2021. Virtual Conference

2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC) | 978-1-7281-1179-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/EMBC46164.2021.9629511

Reinforcement Learning-based Kalman Filter for Adaptive Brain Control in Brain-Machine Interface*

Xiang Zhang, *Student Member, IEEE,* Zhiwei Song, and Yiwen Wang, *Senior Member, IEEE*

***Abstract*— Brain-Machine Interfaces (BMIs) convert paralyzed people's neural signals into the command of the neuro-prosthesis. During the subject's brain control (BC) process, the neural patterns might change across time, making it crucial and challenging for the decoder to co-adapt with the dynamic neural patterns. Kalman Filter (KF) is commonly used for continuous control in BC. However, if the neural patterns become quite different compared with the training data, KF needs a re-calibration session to maintain its performance. On the other hand, Reinforcement Learning (RL) has the advantage of adaptive updating by the reward signal. But it is not very suitable for generating continuous motor states in BC due to the discrete action selection. In this paper, we propose a reinforcement learning-based Kalman filter. We maintain the state transition model of KF for a continuous motor state prediction. At the same time, we use RL to generate the action from the corresponding neural pattern, which is then used as a correction for the state prediction. The RL's parameters are continuously adjusted by the reward signal in BC. In this way, we could achieve a continuous motor state prediction when the neural patterns have drifted across time. The proposed algorithm is tested on a simulated rat lever-pressing experiment, where the rat's neural patterns have drifted across days. Compared with pure KF without re-calibration, our algorithm could follow the neural pattern drift in an online fashion and maintain good performance.**

***Clinical Relevance*— The proposed method bridges the gap between the online parameter adaptation and the continuous control of the neuro-prosthesis. It is promising to be used in adaptive brain control applications during clinical usage.**

I. Introduction

Brain-Machine Interfaces (BMIs) are designed to help disabled people restore their motor functions [1]. The decoder in BMI could translate the subject's neural signal into the movement of the prosthesis. The prosthesis could fulfill the subject's movement intention as a replacement for the real limb. In this way, the subject could Brain Control (BC) the neuro-prosthesis with BMI.

During the BC process, the neural patterns of the subject might be different across time [2]–[4]. One of the typical phenomena is the change of the neural tuning curve. The tuning curve of a neuron is the average firing rate at each motor state of the prosthesis. When the tuning curve becomes different, the decoder needs to capture the change, otherwise the performance would drop. Kalman Filter (KF) is one of the commonly used tools that translates the brain signals into the motor states, which exhibits good performance for continuous motor state prediction in many BMI scenarios [5]–[7]. However, when the neural patterns drift from the training data, KF cannot maintain good performance. It needs a re-calibration session to collect the newest neural data to re-train parameters, which is time-consuming.

Reinforcement Learning (RL) [8]–[11] is an alternative BC algorithm in BMI. It translates neural patterns to motor states and can be adapted in an online fashion. If the action drives the prosthesis closer to the target, the neural-action mapping is reinforced by the reward signal. Otherwise, the mapping is punished. The reward signal could be generated continuously during BC so that the RL model could adaptively update its parameters to follow the change in the neural patterns. However, one of the drawbacks of RL is that the generated action is discrete, which is not very suitable for continuous BC.

In this paper, we propose a reinforcement learning-based Kalman filter. We keep the same state transition function as KF to inherit the advantage of the continuous state estimation. At the same time, based on the current neural patterns, we use RL to predict the corresponding action. The action is then used as an adjustment for the posterior motor state estimation. During the BC process, the RL's parameters are adaptively changed according to the reward signal. In this way, we could have a timely updated RL model to adjust the final motor state prediction, which could potentially follow the neural pattern drift during BC.

We generate a simulated experiment where the rat performs a lever-pressing task. During the experiment, the rat's neural pattern has drifted across days. We tested our proposed algorithm and pure KF on the simulation data to see if our algorithm can follow the change of neural patterns across days. The rest of the paper is organized as follows. Section II provides the mathematical details for Kalman Filter, the adopted RL algorithm, and the proposed algorithm. Section III presents the simulation setup and results. Finally, the conclusion is given in section IV.

*This work is supported by grants from Shenzhen-Hong Kong Innovation Circle (Category D) (No. SGDX2019081623021543), the National Natural Science Foundation of China (No. 61836003), Sponsorship Scheme for Targeted Strategic Partnership (FP902), special research support from Chao Hoi Shuen Foundation, Seed fund of the Big Data for Bio-Intelligence Laboratory (Z0428) from HKUST.

Xiang Zhang is with the Department of Electronic and Computer Engineering, the Hong Kong University of Science and Technology, Hong Kong (email: xzhangaz@ust.hk).

Zhiwei Song is with the Department of Electronic and Computer Engineering, the Hong Kong University of Science and Technology, Hong Kong (email: zsongah@connect.ust.hk).

Yiwen Wang is with the Department of Electronic and Computer Engineering, also with the Department of Chemical and Biological Engineering, the Hong Kong University of Science and Technology, Hong Kong. Yiwen Wang serves as the corresponding author (phone: 852-2358-7053; fax: 852-2358-1485; e-mail: eewangyw@ust.hk).

II. ALGORITHM

The overall structure of the proposed algorithm is shown in Fig. 1, which is a state-observation model. x_t represents the actuator's states (e.g. cursor's position). For each time step, we have an initial estimation of the current state based on the historical state as shown in the first row. In the last row, z_t represents the corresponding neural firing pattern (observation) from the subject. When the actuator's state changes, the neural firing will change accordingly. The neural firing is translated to the motor state through the reinforcement learning model, which is denoted in the middle row. The parameters of the RL model are adaptively changed according to the reward signal r_t at each time step so that it could follow the neural pattern drift across time. Finally, by combining the prior state estimation and the posterior state estimation from the RL model, we could have an adaptive and continuous prediction about the motor state. In the following sub-sections, we will introduce KF and the adopted RL separately. And our proposed algorithm is introduced in the final sub-section.

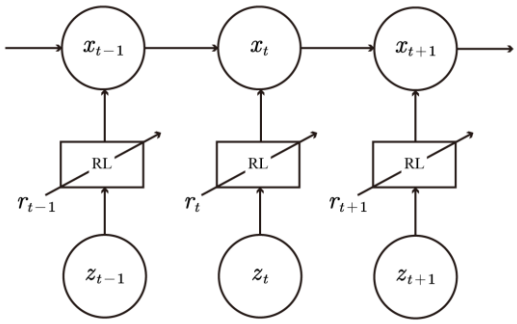

Figure 1. The overall structure of the proposed algorithm

A. State Model

In the state model, we inherit the state transition function of KF. At the current time step t, we denote the motor state as a vector $x_t \in R^{D_x \times 1}$, where D_x is the dimension of the state. The state model is shown as follows

$$x_t = Fx_{t-1} + q. \tag{1}$$

F is the state transition matrix. q is the noise of the state transition model. We assume that q is Gaussian white noise with zero mean, and its covariance matrix is Q. We denote the covariance matrix of the motor state as P_t. The prior update for the state model is shown as follows

$$x_{t|t-1} = Fx_{t-1|t-1} \tag{2}$$

$$P_{t|t-1} = FP_{t-1|t-1}F^T + Q. \tag{3}$$

$x_{t-1|t-1}$ and $P_{t-1|t-1}$ are the posterior estimations of the mean and covariance of the motor state at time $t-1$ respectively.

B. Reinforcement Learning Model

In the observation model, we adopt Attention-Gated Reinforcement Learning (AGREL) [12] as the RL scheme. AGREL can efficiently explore the mapping between the neural pattern and the actuator's state. It employs a three-layer neural network structure. The input layer has D_z nodes, where D_z is the total number of neurons. The hidden layer has H nodes with the sigmoid activation function. The output layer has N nodes which represent the possible actions for the BMI actuator. The final action a_t is selected probabilistically by the softmax policy based on the action value. The motor state after the action selection is denoted as $g_t(z_t)$. If the action drives the motor state closer to the target, a reward $r_t = 1$ will be given to the algorithm, and $r_t = 0$ otherwise. The reward signal is then used to update the weights that connect to the chosen action. More details of AGREL could be found in [12].

C. Reinforcement Learning-based Kalman Filter

In this sub-section, we will show our proposed method that combines the observation mode (AGREL) into the state model. Based on the non-linear neural-state mapping $g_t()$ from AGREL, the observation function is defined in the following

$$g_t(z_t) = x_t + p. \tag{4}$$

p is the noise of the observation model. We assume that p is Gaussian white noise with zero mean, and its covariance matrix is R. The posterior update equations are shown in the following

$$K = P_{t|t-1}\left(P_{t|t-1} + R\right)^{-1} \tag{5}$$

$$x_{t|t} = x_{t|t-1} + K(g_t(z_t) - x_{t|t-1}) \tag{6}$$

$$P_{t|t} = (I - K)P_{t|t-1}. \tag{7}$$

where I is the identity matrix.

The proposed algorithm works as follows. First, we need to set an initial value for the state (x_0 and P_0). For each time step, we calculate the prior motor state through (2-3) and the posterior motor state through (5-7). By comparing the distance from the current state to the target, we can get the reward signal r_t. The reward signal will be then used to update the parameters of AGREL so that $g_t()$ will change accordingly. In this way, our proposed algorithm will have the potential to follow the neural pattern changes during the continuous BC process in BMI.

III. SIMULATION

In this section, we test our algorithm on a simulated experiment, where the rat is performing a lever-pressing task. The rat's neural firing pattern will be different across days. We compare our algorithm with Kalman filter to demonstrate that our algorithm could adaptively follow the neural firing changes without re-calibration sessions.

A. Simulation Setup

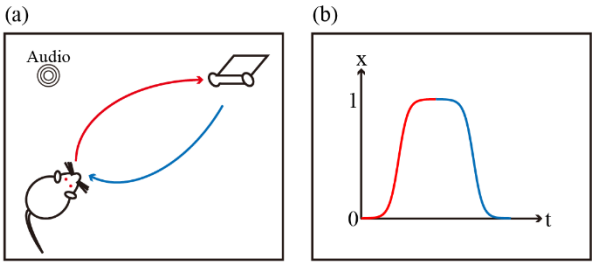

Figure 2. The illustration of the simulated rat lever-pressing task. (a) The physical movement trajectory for the rat. (b) The converted motor state over time.

The simulated lever-pressing task for rats is shown in Fig. 2. In Fig. 2(a), the rat needs to wait at the starting position. When the rat hears the audio cue indicating that the trial starts, the rat needs to get close to the lever, press, and hold it for a period of time. Then the rat will come back to the starting position to get the reward. After a pre-defined inter-trial time, another audio cue will be given and the next trial starts. In Fig. 2(b), the rat's physical trajectory is translated into a one-dimensional motor state $x_t \in R^{D_x \times 1}$ $(D_x = 1)$. At the beginning of the trial, we assume that the motor state is around 0. We denote this stage as the rest stage. When the rat is approaching the lever, the motor state is rising from 0 to 1 as shown by the red curve. During the holding period on the lever, the motor state is maintained at around 1. We denote this stage as the press stage. When the trial is successful and the rat comes back to get the reward, the motor state goes from 1 to 0 as shown by the blue curve.

Based on the trajectory of the motor state, we have generated 4 simulated neurons with different tuning properties. We choose the following tuning function in [13] to simulate the neural firing rate λ for each neuron

$$\lambda = e^{-1+\alpha x_t + \beta v_{x_t}} + \epsilon. \quad (8)$$

α is the tuning parameter related to the motor state x_t. β is the tuning parameter related to the velocity v_{x_t} of the motor state. ϵ is the zero-mean Gaussian noise $\epsilon \in \mathcal{N}(0,0.001)$. The detailed parameters of each simulated neuron are shown in Table I.

Based on the tuning function, we generated the spike of each neuron every 100 ms. And we also binned 4 windows of the historical spike count from each neuron. The dimension of the final observation vector z_t is $D_z = 20$. During the behavioral training across days, we assume that the tuning parameter α of neuron 1 is decreased by 0.1 every day, while other neurons stay the same. For each day, we have 5 trials and each trial lasts for 8 seconds. The Principal Component Analysis (PCA) of the neural patterns over 3 days is shown in Fig. 3. The horizontal and vertical axes represent the neural pattern projections on the first and second principal components respectively. The circles represent the neural patterns at rest or press stage. From the tuning function, we can deduce that the gradual drift of firings of neural 1 only affects the neural patterns at the press stage. The press patterns have a slight drift every day, which is shown by the diamond circle from red, to blue, to green. On the other hand, the neural patterns for the rest stage stay almost the same as shown by the circle with stars.

In this simulation, the input unit number is $D_z = 20$. The hidden unit number is $H = 4$. The number of output units is $N = 2$, where one node represents the rest stage $(a_t = 1)$ and another node represents the press stage $(a_t = 2)$. Given the binned neural pattern z_t, the motor state prediction from AGREL is defined as $g_t(z_t) = P(a_t = 1) \cdot 0 + P(a_t = 2) \cdot 1$, which is then used to update the posterior motor state. The learning rate of AGREL is $\gamma = 0.1$. The parameter for the softmax policy is $\alpha = 3$.

TABLE I. PARAMETERS FOR SIMULATED NEURONS

	Neuron 1	Neuron 2	Neuron 3	Neuron 4
α	0.5	-0.5	0	0
β	0	0	0.5	-0.5

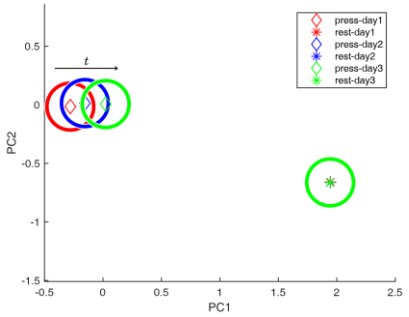

Figure 3. The Principal Component Analysis (PCA) for the neural patterns across days.

B. Simulation Results

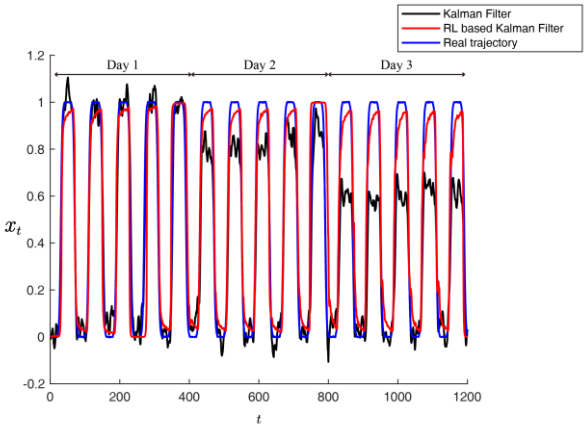

Figure 4. The reconstructed trajectory from Kalman Filter and proposed RL-based Kalman Filter.

We run both KF and RL-based KF on the simulated data. We first train both algorithms on the data of day 1 based on the simulated neural firings and the trajectory. Then we continuously run both algorithms on day 2 and day 3 without a specific re-calibration session. Our algorithm exhibits the advantage of continuously updating the parameters through the reward signal. The reconstructed trajectory is shown in Fig.

4. The x-axis represents the time and the y-axis represents the motor state x_t. The blue curve shows the ground truth trajectory in the simulation. The black and red curves show the results of KF and the proposed algorithm respectively. On day 1, since both algorithms are trained on the same data, the performances are similar. When switching to day 2, since the neural patterns for the press stage have drifted from day 1 as shown in Fig. 3, we can see that KF with fixed parameters cannot follow the neural pattern changes, making the reconstructed trajectory for the press stage around 0.8. On the other hand, for our proposed RL-based KF, due to the parameter adaptation by the reward signal, our algorithm could follow up the neural pattern changes directly from the first trial on day 2. Switching to day 3, the neural patterns for the press stage drift further away from day 1 and the KF's reconstructed trajectory for the press stage can only reach approximately 0.6, whereas our algorithm can still maintain as good performance as during the previous two days. The final performance of both algorithms on day 2 and day 3 is shown in Table II. We can see that both KF and our algorithm reach a similar Correlation Coefficient (CC) to capture the trend of the trajectory. At the same time, our proposed algorithm has a much smaller Mean Square Error (MSE) compared with KF (63% smaller). When facing the neural pattern drift during BC, our algorithm could reconstruct the trajectory more accurately without special re-calibration.

TABLE II. PERFORMANCE FOR DAY 2 AND 3

	KF	**RL based KF**
Correlation Coefficient	0.9629	0.9634
Mean Square Error	0.0405	0.0151

IV. CONCLUSION

BMIs allow the patients to brain control the neuro-prosthesis as part of their bodies. During the BC process in BMI, the subject's neural patterns might be different across time. KF cannot follow the neural pattern changes during usage. On the other hand, RL could follow the neural pattern change but the motor state prediction is relatively rough. In this paper, we combine RL and KF into one state-observation model to take advantage of both algorithms. We keep the state transition function to inherit the continuous state estimation as KF. We incorporate RL into the observation function as a correction for the state estimation. During the BC process, the parameters of RL are adaptively changed according to the reward signal. The changes in neural patterns could be captured by the continuous parameter update. We tested the proposed algorithm on a simulated rat lever-pressing experiment, where the rat's neural patterns vary across days. The KF with fixed parameters cannot maintain good performance after the training day. Our proposed algorithm showed a quick adaptation to new neural patterns, which achieves good performance during the BC process after day 1. It demonstrates that the proposed algorithm is a promising candidate for adaptive and continuous brain control in BMI.

REFERENCES

[1] M. A. Lebedev and M. A. L. Nicolelis, "Brain-Machine Interfaces: From Basic Science to Neuroprostheses and Neurorehabilitation," *Physiological Reviews*, vol. 97, no. 2, pp. 767–837, Mar. 2017, doi: 10.1152/physrev.00027.2016.

[2] A. L. Orsborn, H. G. Moorman, S. A. Overduin, M. M. Shanechi, D. F. Dimitrov, and J. M. Carmena, "Closed-Loop Decoder Adaptation Shapes Neural Plasticity for Skillful Neuroprosthetic Control," *Neuron*, vol. 82, no. 6, pp. 1380–1393, 2014, doi: https://doi.org/10.1016/j.neuron.2014.04.048.

[3] Y. Wang *et al.*, "Tracking Neural Modulation Depth by Dual Sequential Monte Carlo Estimation on Point Processes for Brain–Machine Interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 8, pp. 1728–1741, Aug. 2016, doi: 10.1109/TBME.2015.2500585.

[4] S. Chen, X. Zhang, X. Shen, Y. Huang, and Y. Wang, "Estimating Neural Modulation via Adaptive Point Process Method in Brain-machine Interface*," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*, Jul. 2020, pp. 3078–3081, doi: 10.1109/EMBC44109.2020.9175240.

[5] L. R. Hochberg *et al.*, "Reach and grasp by people with tetraplegia using a neurally controlled robotic arm," *Nature*, vol. 485, pp. 372–375, May 2012.

[6] V. Gilja *et al.*, "A high-performance neural prosthesis enabled by control algorithm design," *Nature Neuroscience*, vol. 15, p. 1752, Nov. 2012.

[7] S. Chen, X. Zhang, X. Shen, Y. Huang, and Y. Wang, "Decoding Transition between Kinematics Stages for Brain-Machine Interface," in IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2019.

[8] J. DiGiovanna, B. Mahmoudi, J. Fortes, J. C. Principe, and J. C. Sanchez, "Coadaptive brain-machine interface via reinforcement learning," *IEEE transactions on biomedical engineering*, vol. 56, no. 1, pp. 54–64, 2009.

[9] Y. Wang, F. Wang, K. Xu, Q. Zhang, S. Zhang, and X. Zheng, "Neural control of a tracking task via attention-gated reinforcement learning for brain-machine interfaces," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, no. 3, pp. 458–467, 2015.

[10] X. Zhang, C. Libedinsky, R. So, J. C. Principe, and Y. Wang, "Clustering Neural Patterns in Kernel Reinforcement Learning Assists Fast Brain Control in Brain-Machine Interfaces," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 9, pp. 1684–1694, Sep. 2019, doi: 10.1109/TNSRE.2019.2934176.

[11] X. Shen, X. Zhang, Y. Huang, S. Chen, and Y. Wang, "Task Learning Over Multi-Day Recording via Internally Rewarded Reinforcement Learning Based Brain Machine Interfaces," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 28, no. 12, pp. 3089–3099, 2020, doi: 10.1109/TNSRE.2020.3039970.

[12] P. R. Roelfsema and A. van Ooyen, "Attention-gated reinforcement learning of internal representations for classification," *Neural computation*, vol. 17, no. 10, pp. 2176–2214, 2005.

[13] W. Truccolo, G. M. Friehs, J. P. Donoghue, and L. R. Hochberg, "Primary Motor Cortex Tuning to Intended Movement Kinematics in Humans with Tetraplegia," *Journal of Neuroscience*, vol. 28, no. 5, pp. 1163–1178, 2008, doi: 10.1523/JNEUROSCI.4415-07.2008.