Cognitive Computation
https://doi.org/10.1007/s12559-020-09742-4

Weakly supervised learning in neural encoding for the position of the moving finger of a macaque

Jingyi Feng[1] · Haifeng Wu[1] · Yu Zeng[1] · Yuhong Wang[1]

Received: 23 July 2019 / Accepted: 10 June 2020

Abstract
The problem of neural decoding is essential for the realization of a neural interface. In this study, the position of the moving finger of a macaque was directly decoded through the neuron spike signals in the motor cortex, instead of relying on the synergy of the related muscle tissues around the body, also known as neural decoding. Currently, supervised learning is the most commonly employed method for this purpose. However, based on existing technologies, unsupervised learning with regression causes excessive errors. To solve this problem, weakly supervised learning (WSL) was used to correct the predicted position of the moving finger of a macaque in unsupervised training. Then, the corrected finger position was further used to train and accurately fit the weight parameters. We then utilized public data to evaluate the decoding performance of the Kalman filter (KF) and the expectation maximization (EM) algorithms in the WSL model. Unlike in previous methods, in WSL, the only available information is that the finger has moved to four areas in the plane, instead of the actual track value. When compared to the supervised models, the WSL decoding performance only differs by approximately 0.4%. This result improves by 41.3% relative to unsupervised models in the two-dimensional plane. The investigated approach overcomes the instability and inaccuracy of unsupervised learning. What's more, the method in the paper also verified that the unsupervised encoding and decoding technology of neuronal signals is related to the range of external activities, rather than having a priori specific location.

Keywords Neural decoding · Macaque moving finger · Unsupervised learning · Weakly supervised learning · Kalman filter · Expectation maximization

Introduction

At present, the realization of brain-like intelligence is essential for the study of the nervous system for the behavior and perception of the external world. Prosthetics, robots, and other devices that fully realize "brain control technology" through psychological means are becoming a reality [1]. For example, tactile mice and wearable technologies can be given to individuals who suffer from comprehensive impairment of the visual and auditory channels. This will enable such individuals to exchange information with the external environment [2]. Additionally, electroencephalogram and electromyography signals have been used to study the nonlinear in the brain to achieve a real-time manual reconstruction system for humeral amputees [3]. These technologies not only have direct positive impacts on the quality of life of impaired users and their abilities to communicate with the environment but also provide a new model of human-computer interaction for both impaired and healthy users [4]. Examples of such applications include computer game control as well as more complex equipment control such as that involved in orthosis, prosthetics, robotic arms, and cognitive robotics [5].

In the process of neural coding, the number of neuron spikes generated when mapping the external world to brain activity is collected by a neural recording instrument to establish a temporal correspondence with external activities. Conversely, the process of neural decoding involves translating neuron spike signals obtained from brain activity into instructions for external motion, which can be used for devices

✉ Haifeng Wu
whf5469@gmail.com

Jingyi Feng
fengjingyione@foxmail.com

Yu Zeng
yv.zeng@gmail.com

Yuhong Wang
1508427758@qq.com

[1] School of Electrical and Information Technology, Yunnan Minzu University, City One, Kunming, China

Published online: 13 July 2020

such as mechanical prostheses. Because the neural coding problem has been discussed in detail in the literature [6] and the human neural coding process itself is being explored constantly and simultaneously [7], we focused on finding a superior solution to the problem of neural decoding for the position of the moving fingers of a macaque.

The moving finger of a macaque was introduced as an early coding problem [8]. A relationship was found between the spike signals of single neurons in the motor cortex and the direction of movement in the upper limbs. Chen et al. presented a linear regression model based on the sparse Bayesian model. They confirmed that the primary motor cortex information recorded from the central anterior gyrus was more abundant than that from the distal end [9]. In the traditional decoding method, earlier supervised learning uses an independent linear method [10], which simplifies the calculation and can track motion trajectory. At present, the state space model (SSM) is more commonly used. Czanner et al. established an SSM model with generalized linear characteristics that is superior to the traditional model in terms of describing the characteristics of neuronal activity [11]. Mehdi et al. expanded upon the implicit SSM model and decoded the neuronal activity recorded from the primary motor cortex of the monkey to perform natural 3D stretching and grasping actions [12]. Rule et al. provided an accurate and efficient means of capturing neural dynamics by combining the autoregressive point process generalized linear models and latent state-space models with point process observations [13]. Among these models, the neural network algorithm has witnessed gradual progress, and the accuracy of the decoding performance of nonlinear algorithms such as time-delay neural network [14] and support vector machine [15] algorithms has been improved considerably.

In general, machine learning can be divided into supervised learning, unsupervised learning, and semi-supervised learning [16]. Supervised learning was proposed earlier and has been extensively developed. In a study on unsupervised learning, unsupervised cubature Kalman filtering decoding (UCKD) [17] was employed to decode the position of the moving finger of a macaque, using the SSM model. Two SSM models were utilized, one for training weights and another for predicting the position of the moving finger of the macaque. This process resulted in a smaller error on the Y-axis, but a larger estimation error on the X-axis. Additionally, in some cases, there was an overall reversal in the Y-axis prediction. In response to the above problems, this paper proposes the use of weakly supervised learning (WSL) [18] to solve the issues of inaccurate estimation of the X-axis and the overall reversal of the Y-axis. In the unsupervised training model, the Kalman filter (KF) and expectation maximization (EM) algorithms were used to decode the position of finger movement; then, WSL was employed to correctly determine the position of the moving finger of the macaque, and finally, the corrected moving position was adopted to train the weight parameters of the model. The experimental results show that the error in the finger movement position decoded using the WSL model is smaller than that in unsupervised learning and is close to that achieved using supervised learning.

The rest of this paper is organized as follows. The "Related work" section provides a clear overview of previous research on unsupervised training of the relevant algorithms. The "Data collection for the moving finger of a macaque" section concisely describes the data collection process. The applied decoding algorithms, including relevant mathematics, are thoroughly discussed in the "Decoding method research" and "Neural decoding method and steps" sections. The "Decoding results and analysis" and "Discussion" sections provide a detailed presentation of the results of each of the experiments using the various algorithms, as well as their significance in the context of the research objectives, limitations of this study, and scope for future work. Finally, the "Conclusion" section concisely summarizes the main contributions of this work.

Related work

Unsupervised training of KF algorithm

In unsupervised training, the SSM model can be used to decode the position of the moving finger of a macaque. The state equation and observation equation can be expressed as [10]

$$y_k = h(y_{k-1}) + w_k \tag{1-a}$$

$$\mathbf{s}_k = \mathbf{f}(y_k) + \mathbf{v}_k \tag{1-b}$$

where

y_k	is the position at time k.
k	0, 1, …K-1.
K	is the total number of sampling points
$\mathbf{s}_k$	is the spike potential signal collected at time k, which is a column vector of $N_e \times 1$.
N_e	is the number of electrodes.
$h(\bullet)$	is the function of a state equation.
$\mathbf{f}(\bullet)$	is the function of an observation equation.
w_k	is Gaussian white noise with a mean of 0 and variance σ^2 [17].
$\mathbf{v}_k = [v_{0k}, v_{1k}, \ldots, v_{N_e-1,k}]^{\mathrm{T}}$	is a Gaussian white noise vector with a mean of 0 and variance matrix $\mathbf{R}_v$ with diagonal $[\delta^2_{0k}, \delta^2_{1k}, \ldots, \delta^2_{N_e-1,k}]$ [17].

In addition, the model requires functions $h(\bullet)$ and $\mathbf{f}(\bullet)$. A common method is to regard the observation function as a

linear function and use the training data to obtain these functions. In this case, Eq. (1) becomes

$$y_k = y_{k-1} + w_k \quad (2-\text{a})$$

$$\mathbf{s}_k = \mathbf{a}_k y_k + \mathbf{b}_k + \mathbf{v}_k \quad (2-\text{b})$$

where $\mathbf{a}_k$, $\mathbf{b}_k$ is the coefficient column vector of $N_e \times 1$.

It should be noted that the SSM model formula in Eq. (2) uses unsupervised training of the KF [10] to predict the finger position, and it requires the weight coefficients $\mathbf{a}_k$ and $\mathbf{b}_k$ in the SSM. In this study, the iterative updating of the weight coefficient was performed using the least squares (LS) approach [19]:

$$\widehat{\mathbf{W}} = \mathbf{Y}^{\dagger}\,\overline{\mathbf{S}} \quad (3)$$

$$R_v = \left(\overline{S} - Y\widehat{W}\right)^T \left(\overline{S} - Y\widehat{W}\right)/(K-H) \quad (4)$$

where

$\mathbf{Y}^{\dagger}$	is the pseudo-inverse of $\mathbf{Y}$.
$\overline{\mathbf{S}} = [\mathbf{S}_H$ $, \mathbf{S}_{H+1}, \ldots, \mathbf{S}_K]^{\mathrm{T}}$	is the matrix set of all spike signal potentials.
K	is the total number of K moments.
H	is the number of H moments ahead of the time correlation [20], which is equivalent to an optimal initial value determined through the preliminary test.

$\mathbf{S}_k = \left[\mathbf{s}_k^{\mathrm{T}}, \mathbf{s}_{k-1}^{\mathrm{T}}, \ldots, \mathbf{s}_{k-H+1}^{\mathrm{T}}\right]^{\mathrm{T}}$, $\mathbf{Y} = [\overline{\mathbf{y}}_k\ , \overline{\mathbf{y}}_{k-1}, \ldots, \overline{\mathbf{y}}_1]^{\mathrm{T}}$ is a matrix set for estimating the position of finger movement, $\overline{\mathbf{y}}_k = [\overline{y}_k, 1]^{\mathrm{T}}$, $\overline{y}_k$ is the estimated position at time k, $\widehat{\mathbf{W}} = [\overline{\mathbf{a}}_k$ $, \overline{\mathbf{b}}_k]^{\mathrm{T}}$ is the estimated coefficient matrix, $\overline{\mathbf{a}}_k$ and $\overline{\mathbf{b}}_k$ are the estimated weights at time k, and $\mathbf{R}_v$ is the observed noise covariance.

Unsupervised training of EM algorithm

The EM algorithm is also based on the SSM principle of Eq. (2). The observation model can be expressed as the Bernoulli probability mass function. Therefore, in the iterative process of the unsupervised EM algorithm, the E-step is defined as follows [21]:

$$y_{k|K} = y_{k|k} + \sigma_{k|k}^2 \left(\sigma_{k+1|k}^2\right)^{-1} \left(y_{k+1|K} - y_{k+1|k}\right) \quad (5)$$

$$R_{k|K} = \sigma_{k|K}^2 + y_{k|K}^2 \quad (6)$$

$$R_{k,k\text{-}1|K} = \sigma_{k\text{-}1,k|K} + y_{k|K} y_{k\text{-}1|K} \quad (7)$$

where $y_{k|j}$ is the mean of state $\overline{y}_k$ with the Gaussian random variable and is calculated using the filtering algorithm ($j = k$) at time k, $\sigma_{k|j}^2$ is the covariance of state $\overline{y}_k$ with the Gaussian random variable and is calculated using the smoothing algorithm ($j = K$) at time k, $R_{k|K}$ and $R_{k,\,k\text{-}1|K}$ are the joint covariances of states $\overline{y}_k^2$ and $\overline{y}_k\ \overline{y}_{k\text{-}1}$, respectively, as calculated using the smoothing algorithm, and $\sigma_{k\text{-}1,\,k|K}$ is the state space covariance of state $\overline{y}_k$ $\overline{y}_{k\text{-}1}$ as calculated using the smoothing algorithm.

Therefore, the expected log likelihood value of the complete data can be determined by calculating the expected value $y_{k|K}$ and covariance $R_{k|K}$, $R_{k,\,k\text{-}1|K}$ of the state variable. The filtering algorithm gives the estimated value of the initial state, and the smoothing algorithm gives the estimated value of the ideal observation state.

In the M-step, maximizing the expected value of the log likelihood of the complete data of Eqs. 5–7, the updated parameters can be defined as follows [21]:

$$\overline{y}_0^{(l+1)} = \frac{1}{2} y_{1|K} \quad (8)$$

where $\overline{y}_0^{(l+1)}$ is the initial state value of the l+1th iterative cycle. To evaluate the expected value of the best complete data log-likelihood in the E-step, the EM algorithm mainly uses the filtering algorithm, the fixed interval smoothing algorithm, the state space covariance algorithm in the E-step of Eqs. 5–7, and the M-steps of Eq. (8) to complete cycle iteration.

The weights and finger positions trained on Eqs. 2–8 are inaccurate. In the next section, we describe the use of WSL to adjust the estimated position, as well as the application of the adjusted moving position of the finger to obtain more reliable weight parameters.

Data collection for the moving finger of a macaque

The specific codec process for the moving finger of the macaque is shown in Fig. 1. Regarding the relevant collection

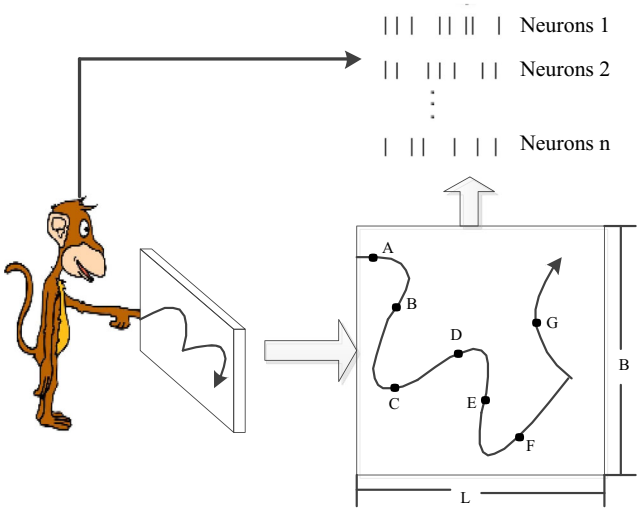

Fig. 1 Track coding of the moving finger of the macaque

equipment, a normal healthy macaque was selected, and the finger could move freely in an area of length L and width B.

First, the macaque was trained, and bright spots appeared as targets in the active area. At different times, the objects appeared in different positions, and the finger of the macaque moved to the target. After repeated training, the macaque could independently complete this task. Secondly, the period of occurrence of the target was defined as Δt, and the implanted neural electrode array related to the moving finger could collect information $\mathbf{s}_k$ at times $k\Delta t$, where $k = 0, 1, ..., K\text{-}1$, etc. Then, it was necessary to record the position of the moving finger at each moment. For example, when the finger moved to A at time $k\Delta t$, the signal $\mathbf{s}_k$ and position y_k were recorded for that moment. Sequential recording was used for position y_k of B, C, D, E, etc. Finally, K groups of data were collected in total. The neural decoding technique was required to obtain $\widehat{y}_k$ through $\mathbf{s}_k$ and to attempt to make the estimated $\widehat{y}_k$ consistent with the actual y_k.

In this study, it was necessary to decode the finger movement position $\widehat{y}_k$ using the neuron spike potential signal in unsupervised learning. WSL was employed to correct the instability of the unsupervised learning and numerous reverse problems. After adjusting $\widehat{y}_k$, we continued to use the adjusted $\widehat{y}_k$ to train weight $\widehat{\mathbf{W}}$ to improve the accuracy of the trained model, which is more conducive to predicting the position of the finger movement track.

Decoding method research

WSL

First, after reviewing several studies and experiments on unsupervised learning, we determined the following rule, i.e., a few predicted values and real values exert a symmetry effect on the median value of the finger activity area. This allows the idea of the conversion to be remembered. In this study, this idea is still considered as the category of WSL [18].

This section mainly discusses the idea of WSL and how it was specifically applied. In addition, the abscissa and ordinate are represented by the position. WSL was utilized in this study, and the plane area of the finger movement was divided into four quadrants. As can be seen from Figs. 2 and 3, the position $z(x_k', y_k')$ was processed in the training data, providing the quadrant location in the defined coordinate system where $i = 1, 2, 3,$ or 4. We only know that the finger has moved to certain areas among the four areas in the plane, and do not know the actual track value; moreover, the approximate area is also the essence of the idea of WSL. Furthermore, an imprecise label of WSL is acquired through a one-to-one conversion of the real position to obtain the areas of the motion track.

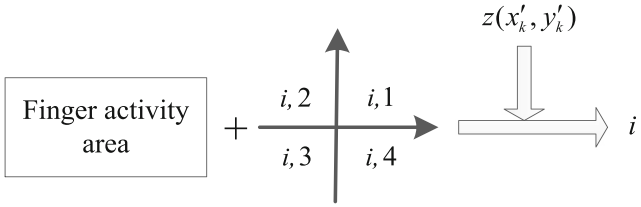

Fig. 2 WSL data processing

Because the previous unsupervised training method could be used to train x_k' and y_k' roughly, most of the trained positions are in the reverse state at the center of the motion region. The idea of WSL was mainly applied to the training data, and x_k' as well as y_k' have only positive values. Therefore, the process can be further analyzed and operated as follows:

1) Creating a virtual X′ and Y′ axis: The maximum value $x'_{\max}, y'_{\max}$ and minimum value $x'_{\min}, y'_{\min}$ are, respectively, obtained from the training data $x_k', k = 1, 2, \ldots, K$ and $y_k', k = 1, 2, \ldots, K$. The virtual Y′-axis is then represented as $\left(x'_{\max} + x'_{\min}\right)/2$, and the X′-axis is represented as $\left(y'_{\max} + y'_{\min}\right)/2$.
2) The abscissa and ordinate of the training position y_k are based on the virtual Y′-axis and X′-axis, respectively. If the true abscissa is more than $\left(x'_{\max} + x'_{\min}\right)/2$, then it is expressed as 1, and if it is less than $\left(x'_{\max} + x'_{\min}\right)/2$, it is expressed as 0. Likewise, if the true ordinate is more than $\left(y'_{\max} + y'_{\min}\right)/2$, then it is expressed as 1 and as 0 otherwise.
3) The estimated abscissa and ordinate of the training data in unsupervised learning are also obtained from the virtual Y′-axis and X′-axis, respectively. If the estimated abscissa is more than $\left(x'_{\max} + x'_{\min}\right)/2$, then it is expressed as 1, and if it is less than $\left(x'_{\max} + x'_{\min}\right)/2$, then it is 0. Likewise, if the estimated ordinate is more than $\left(y'_{\max} + y'_{\min}\right)/2$, then it is expressed as 1. Otherwise, it is expressed as 0.
4) Judgment: The abscissa and ordinate estimated by unsupervised learning have a one-to-one correspondence with the true abscissa and true ordinate. For example, the estimated and real abscissa are simultaneously 1 or 0 or vice versa, which is judged as positive. The judgement is similar for the estimated and real ordinate.

Finally, if the estimated abscissa and ordinate are determined to be positive, the corresponding position remains unchanged. If the judgment is reversed, the abscissa or ordinate should be overturned on the virtual Y′-axis and the X′-axis, which can be expressed as follows:

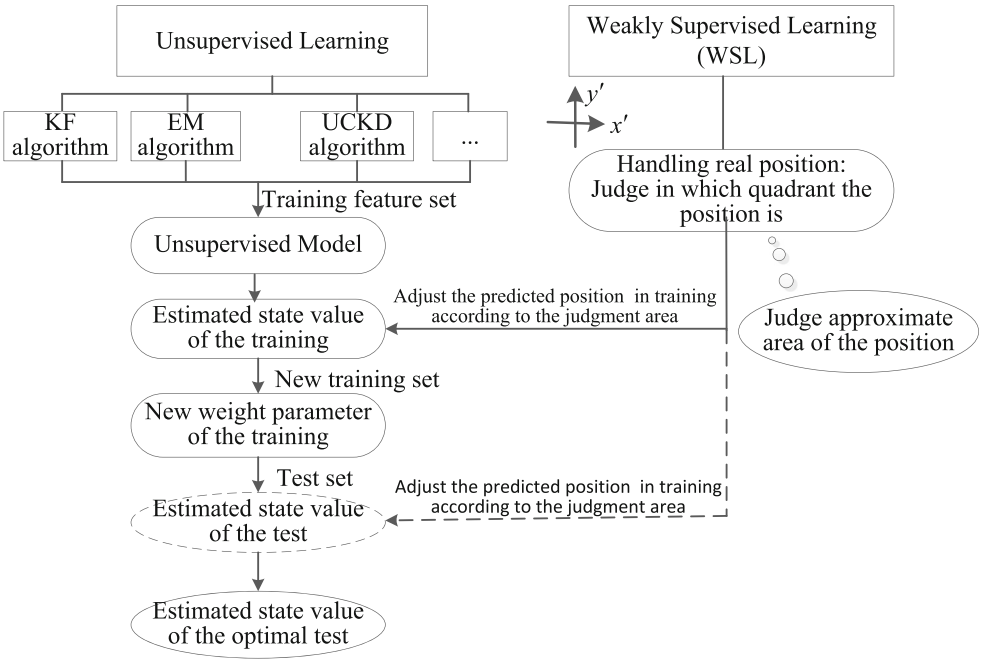

Fig. 3 WSL decoding diagram

$$\overline{y}'_k = \frac{(y_{\max} - y_{\min})\left(\overline{y}_k - \overline{y}_{\min}\right)}{\overline{y}_{\max} - \overline{y}_{\min}} + y_{\min}$$
$$\widetilde{y}_k = 2\left(\mathrm{Y} - \overline{y}'_k\right) + \overline{y}'_k \tag{9}$$

where $y_{\max}$ and $\overline{y}_{\max}$ are the maximum values in the positions before and after unsupervised training respectively. Similarly, $y_{\min}$ and $\overline{y}_{\min}$ are the minimum values. Y is represented as $(y_{\max} + y_{\min})/2$. $\overline{y}'_k$ is the linearly processed position of $\overline{y}_k$ after unsupervised training, and $\widetilde{y}_k$ is the estimated training position after WSL.

WSL's training and decoding of finger movement position via KF algorithm

After establishing the WSL model, the predicted values and corresponding peak values of the original neurons constitute a new supervised mode, which require further training to optimize the weight of the model.

Weakly supervised processing of the unsupervised KF algorithm is required, as discussed in the "Unsupervised training of KF algorithm" section. Subsequently, the weight calculation for the SSM model can be performed using the LS approach [19]:

$$\widetilde{\mathbf{W}} = \widetilde{\mathbf{Y}}^{\dagger}\, \overline{\mathbf{S}}' \tag{10}$$

$$\mathbf{R}_v = \left(\overline{\mathbf{S}}' - \widetilde{\mathbf{Y}}\widetilde{\mathbf{W}}\right)^{\mathrm{T}}\left(\overline{\mathbf{S}}' - \widetilde{\mathbf{Y}}\widetilde{\mathbf{W}}\right)/(K-H) \tag{11}$$

where $\widetilde{\mathbf{Y}}$ is the position $\mathbf{Y}$ after WSL, $\overline{\mathbf{S}}' = \left[\mathbf{S}'_H, \mathbf{S}'_{H+1}, \ldots, \mathbf{S}'_K\right]^{\mathrm{T}}$, $\mathbf{S}'_k = \left[\mathbf{s}'^{\mathrm{T}}_k, \mathbf{s}'^{\mathrm{T}}_{k-1}, \ldots, \mathbf{s}'^{\mathrm{T}}_{k-H+1}\right]^{\mathrm{T}}$, $\mathbf{s}'_k$ is the spike signals of the test data at the time k and is mutually exclusive with the training data $\mathbf{s}_k$, and $\widetilde{\mathbf{W}}$ is the estimated matrix after WSL.

In the test, according to the weight $\widetilde{\mathbf{W}}$, it is necessary to estimate the finger movement position $\widehat{y}_k$ from the given neuron spike signals. Using the KF algorithm [12]

$$\mathbf{G}_k = P_{k\text{-}1}\widetilde{\mathbf{W}}^{\mathrm{T}}\left(\widetilde{\mathbf{W}}P_{k\text{-}1}\widetilde{\mathbf{W}}^{\mathrm{T}} + \mathbf{R}_v\right)^{-1} \tag{12}$$

$$\boldsymbol{\alpha}_k = \mathbf{S}'_k - \widetilde{\mathbf{W}}\widehat{y}_{k\text{-}1} \tag{13}$$

$$\widehat{y}_k = \widehat{y}_{k\text{-}1} + \mathbf{G}_k\boldsymbol{\alpha}_k \tag{14}$$

$$P_k = P_{k\text{-}1} - \mathbf{G}_k\widetilde{\mathbf{W}}P_{k\text{-}1} + \sigma^2 \tag{15}$$

where P_k is the covariance of the estimated $\widehat{y}_k$ at time k, $\mathbf{G}_k$ is the Kalman gain vector at time k, and $\widehat{y}_k$ is the estimated position of the test data at time k.

WSL's training and decoding of finger movement position via EM Algorithm

Similarly, according to the establishment of the WSL model, the trained position of the unsupervised EM algorithm is processed. After processing, the EM algorithm is used to estimate the weight of the finger movement position, which can be expressed as

$$\widetilde{\mathbf{a}} = \frac{K\sum_{k=1}^{K}\mathbf{S}'_k\widetilde{y}_k - \sum_{k=1}^{K}\widetilde{y}_k\sum_{k=1}^{K}\mathbf{S}'_k}{\sum_{k=1}^{K}\left(\widetilde{y}_k^2 + P_k\right) - \sum_{k=1}^{K}\widetilde{y}_k} \tag{16}$$

$$\widetilde{\mathbf{b}} = \frac{1}{K}\left(\sum_{k=1}^{K}\mathbf{S}_k' - \overline{\mathbf{a}}\sum_{k=1}^{K}\widetilde{y}_k\right) \tag{17}$$

$$\widetilde{\mathbf{W}} = \left[\widetilde{\mathbf{a}}, \widetilde{\mathbf{b}}\right]^{\mathrm{T}} \tag{18}$$

where

$\widetilde{\mathbf{a}}$ and $\widetilde{\mathbf{b}}$ are the matrices estimated after WSL.

In addition, the iterative update calculation for the observed noise covariance $\mathbf{R}_v$ takes the form of Eq. 4 or 11.

In the test, according to the weight $\widetilde{\mathbf{W}}$, the finger movement position $\widehat{y}_k$ was estimated from the given neuron spike signals. The experimental principle of the filtering algorithm is described in Eqs. 5–8, wherein the maximum expected value is found using the E-step and M-step loop iterations.

Finally, we used the KF and EM algorithms based on the WSL principle to estimate the position $\widehat{y}_k$ by comparing the neuron spike signals to the actual finger position y_k.

Neural decoding method and steps

Data sources

The data were collected from the Hatsopoulos Laboratory [22] and can be found at https://booksite.elsevier.com/9780123838360/. The collected dataset is called the "hatsopoulos dataset." For a description of the data collected, please refer to the "Data collection for the moving finger of a macaque" section, where the parameter value set was as follows.

1) Active area length L = 25 cm, width B = 18 cm
2) Number of electrodes N_e = 42
3) Sampling period Δt = 70 ms
4) Data length K = 3101

Two groups of experimental data were collected, and the relevant parameters of each group of data were set as listed above. The specific characteristics of the data are as follows.

1) Data 1: This dataset consists of a neuron feature matrix of dimensions $K \times N_e$ and a position tag matrix of dimensions $K \times 2$. The first column in the tag matrix is the X-axis, and the second column is the Y-axis.
2) Data 2: The format is the same as that of Data 1, but some of the sample data are outside the active range.

Data 1 and Data 2 were collected under the same conditions. The difference is that Data 2 not only has continuous finger movement in a random direction but also has horizontal or vertical movement for a period of time. In the experiments, the sample points beyond the active area were removed.

Experiment and parameter setting

It should be noted that all the WSL experiment results were combined with the time correlation parameter H ($H > 1$) [20], so that the error would be further reduced relative to the case without introducing the parameter. The specific H-value is discussed in the "Decoding results and analysis" section. It is also noteworthy that some traditional algorithms do not introduce H or have $H = 1$.

The lengths of Data 1 and Data 2 were both K = 3101. Data 1 and Data 2 were divided into five groups of experiments to judge the neural decoding performance of WSL. The specific experiments were as follows.

1) *Experiment A*: Holdout verification was applied to both Data 1 and Data 2 [23]. In each case, 70% of the hatsopoulos dataset was used for training, and 30% was used for testing, which can intuitively represent the decoding trajectory.
2) *Experiment B*: Data 1 was tested using M-fold cross-validation [23], taking M = 10.
3) *Experiment C*: Data 2 was tested using M-fold cross-validation [23], taking M = 10.
4) *Experiment D*: Data 1 was used as the training sample to obtain the training model, and Data 2 was input into the model for testing.
5) *Experiment E*: Data 2 was used as the training sample to obtain the training model, and Data 1 was input into the model for testing.

The error between the decoded and actual positions in *Experiments B* and *C* is the root mean square error e_c, which can be calculated as follows:

$$e_c = \frac{1}{M}\sum_{m=1}^{M}\sqrt{\frac{1}{K'}\sum_{k=0}^{K'-1}\left(\widehat{y}_{m,k} - y_{m,k}\right)^2} \tag{19}$$

where the position decoded during the M-th cross-validation is $\widehat{y}_{m,k}$, the actual position is $y_{m,\,k}$, and the data length of the cross-validation is K'.

The error between the decoded and actual positions in *Experiments D* and *E* is the root mean square error e_r, which can be calculated as follows:

$$e_r = \sqrt{\frac{1}{K}\sum_{k=0}^{K-1}\left(\widehat{y}_k - y_k\right)^2} \tag{20}$$

where the decoded position is $\widehat{y}_k$ and the actual position is y_k in the test data.

Next, the error in the two-dimensional plane is the root mean square error e_{xy}, which can be represented as follows:

$$e_{xy} = \sqrt{\frac{1}{K}\sum_{k=0}^{K-1}\left(\widehat{x}_k' - x_k'\right)^2 + \left(\widehat{y}_k' - y_k'\right)^2} \tag{21}$$

where the abscissa and ordinate decoded in the test data are $\hat{x}'$ and $\hat{y}_k'$, respectively, and the true abscissa and ordinate are x_k' and y_k', respectively.

The above five groups of experiments involved calculating e_c and e_r using linear [10] and KF [10] algorithms with supervised learning, UCKD with unsupervised learning, and the KF and EM algorithms with WSL. The relevant parameters were set as follows.

1) Linear: The weight between the finger movement position and spike signals was trained using the LS approach and then decoded with the test data.
2) KF: The LS method was used to train the weight between the spike signals and finger movement position. The initial value $y_0 = 0$, covariance $P_{y,\,0|0} = 10$, state noise variance $Q_w = 0.8$, and observed noise variance $\mathbf{R}_v = (\bar{\mathbf{s}} - \hat{\mathbf{s}})^{\mathrm{T}} (\bar{\mathbf{s}} - \hat{\mathbf{s}}) / (K-1)$, where $\bar{\mathbf{s}} = [\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{K-1}]^{\mathrm{T}}$ and $\hat{\mathbf{s}} = \mathbf{Y}^{\dagger}\widehat{\mathbf{W}}$.
3) XGBoost: The initial max_depth = 5, learning_rate = 0.1, n_estimators = 100, objective = 'reg:linear', n_jobs = −1, eval_metric = 'logloss', and verbose = 100.
4) LightGBM: The initial num_leaves = 31, learning_rate = 0.05, n_estimators = 100, eval_metric = 'logloss', and verbose = 100.
5) UCKD: The initial decoded position $y_0 = 10$, covariance $P_{y,\,0|0} = 10$, and state noise variance $Q_w = 0.8$. The observed noise variance $\mathbf{R}_v = \mathrm{diag}(\mathrm{ones}(1, 42))$ represents generation behavior 1, and the diagonal matrix is listed as 42. The initial weight $\widetilde{\mathbf{W}}_0 = \mathrm{ones}(1, 84)$ represents generation behavior 1, and the all-one matrix is listed as 84. The covariance $\mathbf{P}_{w,\,0|0} = \mathrm{diag}(\mathrm{ones}(1, 84))$represents generated behavior 1, the diagonal matrix is listed as 84, the weighted noise $\mathbf{R}_w$=ones(1, 84) represents generation behavior 1, and the all-ones matrix is listed as 84. The forgetting factor λ= 0.005.
6) U-EM: This is the EM algorithm in unsupervised learning. The initial decoded position y_0= 10, covariance$P_{y,\,0|0} = 10$, and system noise variance R_w= 2. The observed noise variance $\mathbf{R}_v$=diag(ones(1, 42)) represents generation behavior 1, and the diagonal matrix is listed as 42. $\widetilde{\mathbf{W}}_0 = \mathrm{ones}(2, 42)$ represents generation behavior 2, and the all-ones matrix is listed as 42.
7) WSL-KF: This is the KF algorithm in WSL. The initial decoded position y_0= 10, covariance $P_{y,\,0|0} = 10$, and system noise variance $R_w = 0.8$. The observed noise variance $\mathbf{R}_v = \mathrm{diag}(\mathrm{ones}(1, 42H))$ represents generation behavior 1, and the diagonal matrix is listed as $42H$. The initial weight $\widetilde{\mathbf{W}}_0 = \mathrm{ones}(2, 42H)$ represents generation behavior 2, and the all-ones matrix is listed as $42H$.
8) WSL-EM: This is the EM algorithm in WSL. The initial decoded position y_0= 10, covariance $P_{y,\,0|0} = 10$, and system noise variance R_w= 2. The observed noise variance$\mathbf{R}_v = \mathrm{diag}(\mathrm{ones}(1, 42H))$ represents generation behavior 1, and the diagonal matrix is listed as $42H$. $\widetilde{\mathbf{W}}_0 = \mathrm{ones}(2, 42H)$ represents generation behavior 2, the all-ones matrix is listed as $42H$.

Decoding steps of the WSL-KF and WSL-EM

In the previous section, the decoding approaches based on the WSL-KF and WSL-EM algorithms were given. The completed decoding steps for the position of the moving finger of the macaque are summarized in Fig. 4.

Decoding results and analysis

Figure 5(a, b) and 5(c, d) shows the positions of the *X*- and *Y*-axes for the 150 sample points in *Experiment A* for Data 1 and 2, respectively. As seen from the figure, Linear, KF, UCKD, WSL-KF, and WSL-EM were selected from the eight algorithms for visual comparison. Except for the unsupervised UCKD, most of these algorithms are capable of tracking the moving trajectory. The linear and KF approach with supervised learning, WSL-KF algorithm, and WSL-EM technique can follow the true curve trajectory, but the algorithms perform differently owing to the difference between supervised learning and WSL. First, the linear algorithm will have more jitter and glitches in the decoding of Data 1 and Data 2, especially on the *Y*-axis for Data 2. The performance is more evident between 100 and 150, where it becomes possible to see the moving position as

an independent state. The supervised KF has a higher decoding accuracy than the linear method, and its tracking effect is significantly better than that of the linear method. Second, as can be seen, most of the moving positions decoded using unsupervised UCKD cannot be employed to track the actual trajectory, and most of the decoded positions are reversed. Furthermore, the WSL-KF is a relatively supervised KF, which may have a slight negative impact on the decoding performance, but the partial points decoded are still superior to the supervised KF approach. For example, where the *X*-value in Data 1 is roughly in the range of 30–70 and the *Y*-value is approximately 50–100, the decoding curve is almost closer to the true position trajectory than that obtained using the supervised KF. Finally, the WSL-EM sometimes tracks the true position almost completely, which is an improvement compared with the other algorithms. Examples of this situation include the *X*-axis range of roughly 30–60 and *Y*-axis range of 70–80 in Data 1, as well

as the *X*-axis range of 100–120 in Data 2, as can be seen in Fig. 5(**a**, **b**) and (**c**, **d**). These results indicate that WSL

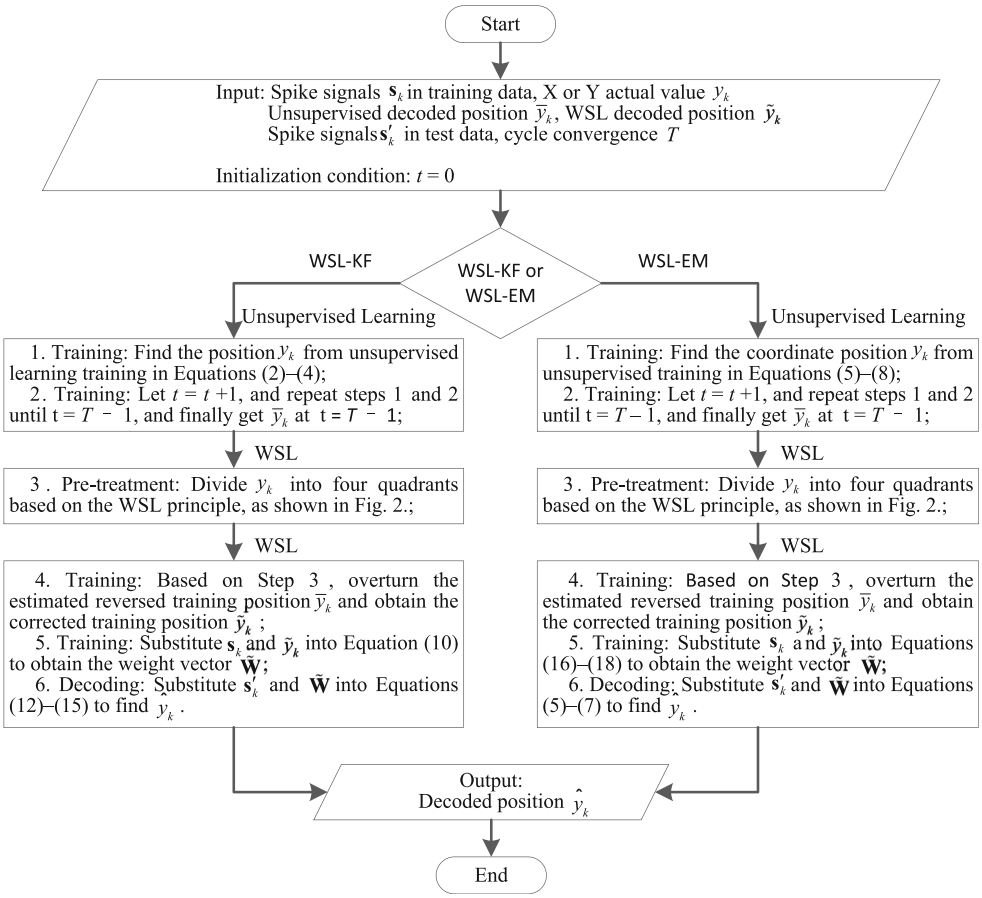

Fig. 4 Neural decoding steps of the WSL-KF and WSL-EM for the macaque finger movement position

sometimes yields better tracking performance than traditional supervised learning.

Table 1 lists the mean squared errors of the *X*- and *Y*-values on the one-dimensional plane for each algorithm from *Experiments B–E*. In descending order, the average errors of the *X*-values for the four groups of experiments are U-EM, UCKD, WSL-KF, linear, KF, LightGBM, XGBoost, and WSL-EM. The average *Y*-value errors in the descending order are UCKD, U-EM, WSL-KF, linear, XGBoost, LightGBM, WSL-EM, and KF. Compared with the XGBoost algorithm, which has the smallest supervised error in the *X*-value, the WSL-EM only differs by approximately 1.6% while exhibiting the lowest decoding error, which is approximately 37.5% lower than the average error obtained when using unsupervised UCKD. Additionally, compared with unsupervised U-EM, which has the smallest unsupervised error in

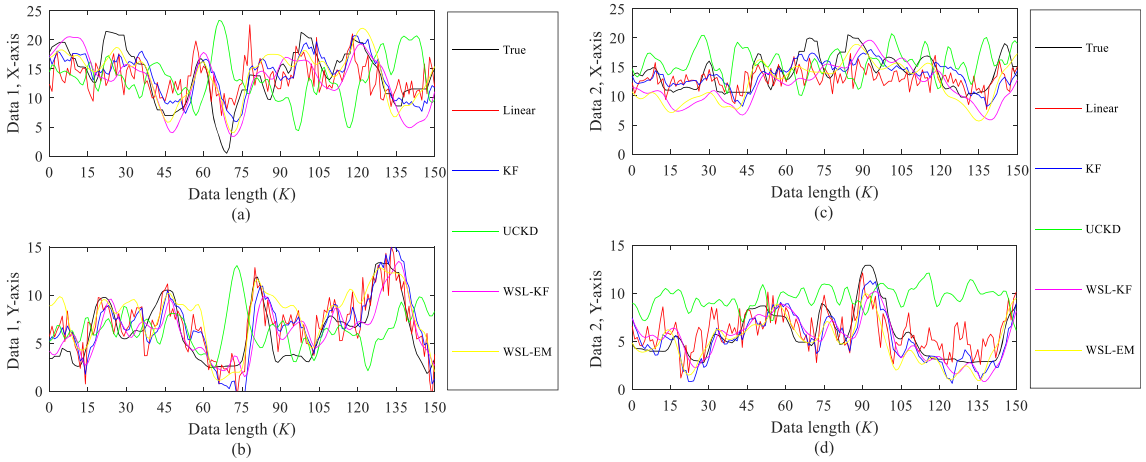

Fig. 5 (**a**, **b**) Curves for decoding finger position in *Experiment A* with Data 1 along the *X*- and *Y*-axes. (**c**, **d**) Curves for decoding finger position in *Experiment A* with Data 2 along the *X*- and *Y*-axes

Table 1 *X*-axis (*Y*-axis) estimation error, unit: CM

Algorithm	*Experiment B*	*Experiment C*	*Experiment D*	*Experiment E*	average (*B*+*C*+*D*+*E*)/4
Linear	3.813 (2.003)	3.898 (2.460)	4.084 (3.010)	3.954 (2.132)	3.937 (2.401)
KF	3.060 (1.498)	3.877 (1.956)	4.569 (2.974)	3.602 (1.649)	3.777 (2.019)
XGBoost	3.646 (1.913)	3.715 (2.307)	3.837 (2.652)	3.758 (2.084)	3.739 (2.239)
LightGBM	3.663 (1.926)	3.709 (2.288)	3.914 (2.524)	3.773 (2.075)	3.765 (2.203)
UCKD	6.422 (3.022)	5.758 (5.656)	5.776 (2.882)	5.596 (4.977)	5.888 (4.134)
U-EM	6.386 (2.642)	6.688 (4.303)	8.328 (4.650)	7.085 (4.326)	7.122 (3.980)
WSL-KF	3.412 (2.218)	3.561 (2.321)	5.421 (3.477)	3.828 (1.879)	4.056 (2.474)
WSL-EM	3.142 (2.069)	3.663 (2.288)	4.518 (2.621)	3.398 (1.675)	3.680 (2.163)

the Y-value, WSL-EM reduces the decoding error in terms of the Y-values by approximately 45.7%. Furthermore, specific to each group of experiments, the mean square errors of the WSL-KF and WSL-EM algorithms are similar to or less than that of the supervised algorithm and are less than that of the unsupervised UCKD and U-EM approaches. The decoding error of the WSL-KF algorithm is less than that of the supervised KF method, in terms of the X-values in *Experiment C*, because the introduced time-dependent parameter *H* has a significant effect on error reduction. The traditional linear, KF, XGBoost, LightGBM, UCKD, and U-EM algorithms do not introduce *H*. The decoding error of the WSL-KF approach is significantly higher in *Experiment D*, indicating that the decoding ability of WSL-KF is weak for the *X*- and *Y*-values in *Experiment D*. The decoding error of UCKD is considerably lower for the *Y*-values in *Experiment D*, indicating that a majority of the positions decoded by the unsupervised UCKD are in the positive state. The average decoding error of the U-EM algorithm is markedly higher for the *X*-values, whereas the average error of the WSL-EM is markedly smaller. This indicates that the U-EM algorithm demonstrates more reverse prediction of trajectory movement, although it is more accurate after reverse processing.

Table 2 lists the mean square errors of the XY axis on the two-dimensional plane for each algorithm, from *Experiments B* to *E*. In a descending order of the average errors for the four groups of experiments, the algorithms are ranked as follows: U-EM, UCKD, WSL-KF, linear, LightGBM, XGBoost, KF, and WSL-EM. The decoding error of WSL-EM is within 0.4% of the best value, which was obtained using the supervised KF approach. When compared with the unsupervised UCKD, the performance of WSL-EM improves by approximately 41.3%. When compared with the unsupervised U-EM, the performance of WSL-EM improves by approximately 47.7%. The decoding results are similar to the conclusions drawn in Table 2. The WSL-EM decoding error is still less than those of the other algorithms. Note that the generalization error of the *Y*-values is generally lower than that of the *X*-values. We believe that some of the test data used for the *Y*-values have more similarities with the training data set, thereby resulting in a stronger overall generalization ability for the *Y*-values as compared with that for the *X*-values.

The finger movement position at the current time is correlated with the neuron spike potential signals at previous moments. Here, the experimental results are presented in terms of the *X*- and *Y*-values of Data 1 and Data 2 from *Experiments B* to *E*. The time correlation *H* and number of iteration cycles for

Table 2 Estimation error of the two-dimensional plane, unit: CM

Algorithm	*Experiment B*	*Experiment C*	*Experiment D*	*Experiment E*	average (B+C+D+E)/4
Linear	4.307	4.609	5.073	4.492	4.620
KF	3.407	4.343	5.452	3.962	4.291
XGBoost	4.117	4.373	4.664	4.297	4.363
LightGBM	4.139	4.358	4.657	4.306	4.365
UCKD	7.098	8.071	6.455	7.489	7.278
U-EM	6.911	7.953	9.538	8.301	8.176
WSL-KF	4.070	4.251	6.440	4.264	4.756
WSL-EM	3.762	4.319	5.223	3.788	4.273

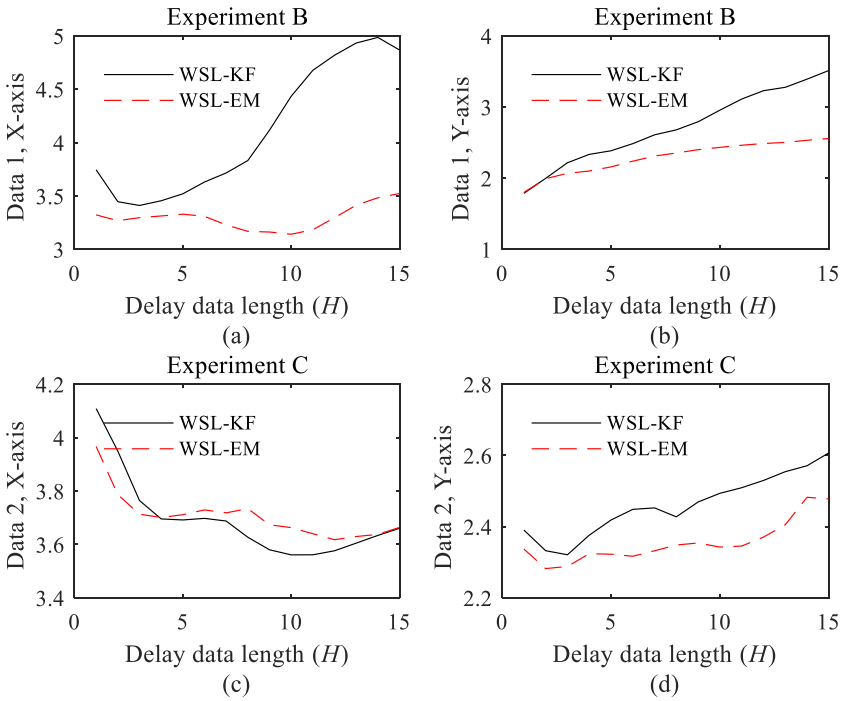

Fig. 6 Estimation error on the *X*- (left) and *Y*-axes (right) vs. temporal correlation parameter *H* in *Experiments* (**a**, **b**) *B* and (**c**, **d**) *C*

convergence were tested to study the effects of the positions decoded using WSL-KF and WSL-EM, respectively.

Fig. 6 shows the effect of the *H*-value on the decoding performance in the WSL model. The mean square error curves of the *X*- and *Y*-values decoded using WSL-KF and WSL-EM in *Experiments B* and *C*, respectively, are presented. The error curve of the *X*-values obtained by applying the WSL-KF to Data 1 shows a downward trend before about $H = 3$, decreasing to about 3.5 cm, which is followed by a rapid upward trend. In addition to this particular case, the error curves of WSL-KF and WSL-EM show downward trends before about $H = 10$, and the error curve is minimized at $H = 10$. The error curve then gradually increases. Meanwhile, the error curves of the *Y*-values obtained by applying WSL-KF and WSL-EM to Data 1 are increasing. For Data 2, the trend is decreasing before $H = 3$, until about 2.3 cm, followed by an increase. *H*-value that is too small will lead to under-fitting, and one that is too large will lead to over-fitting, resulting in poor generalization ability. Of course, we think the reason that the *Y*-value error increased for Data 1 is that the *Y*-value decoding error could reach levels as low as 1.7 cm. It is obvious that continuing to increase the *H*-value does not have much impact. However, the increased error range at $H = 3$ is negligible within 0.3 cm, so the time correlation can still be taken as $H = 3$ for the *Y*-values of Data 1.

In addition, we used Data 1 and Data 2 to test the effects of the time-dependent *H* of the WSL-KF and WSL-EM algorithms in *Experiments D* and *E*, as shown in Fig. 7. Except for the decoding error of WSL-EM being too large at $H = 3$ on the Y-axis for Data 1 and data 2, the decoding errors of the other algorithms have little effect, despite some differences, and satisfy the *H*-value requirements mentioned above for the X- and Y-axis tests. Here, the most significant experimental results obtained from tests are given, using the Y-values of Data 1 for training and those of Data 2 for testing in *Experiment D*. The time correlation *H*-value of the WSL-EM algorithm was taken as 8. The above analysis shows that except for two special cases, the time correlations of X- and Y-axis decoding are $H = 3$ and $H = 8$, respectively. The time correlations of the *X*- and *Y*-values are $H = 10$ and $H = 3$, which can more effectively ensure that the WSL algorithm has a small decoding error for the finger movement position.

Figure 8 shows the effect of the number of iteration cycles *T* on the decoding performance in the WSL model. The mean square error curves of the *X*- and Y-axes decoded using WSL-KF and WSL-EM from *Experiments B* and *C*, respectively, are presented. It can be seen from Fig. 8 that although the error curve of the *X*-values of Data 1 first rises and then becomes stable, the mean square error curve of the finger movement position tends to become stable after $T = 3$ in the WSK-KF results. In the WSL-EM results, the error curve of the *X*-values for Data 1 first rises, then decreases, and subsequently increases until about $T = 60$, after which it essentially becomes stable. The error curves for the other cases are stable after $T = 8$. We think that the error increases with the number of the iteration cycles in the experimental results because *M*-fold cross-validation was used. Decoding error also occurs in unsupervised learning under these parameter settings, and the errors are too large in the *M* cross-validation results, leading to instability in the mean error. This characteristic indicates

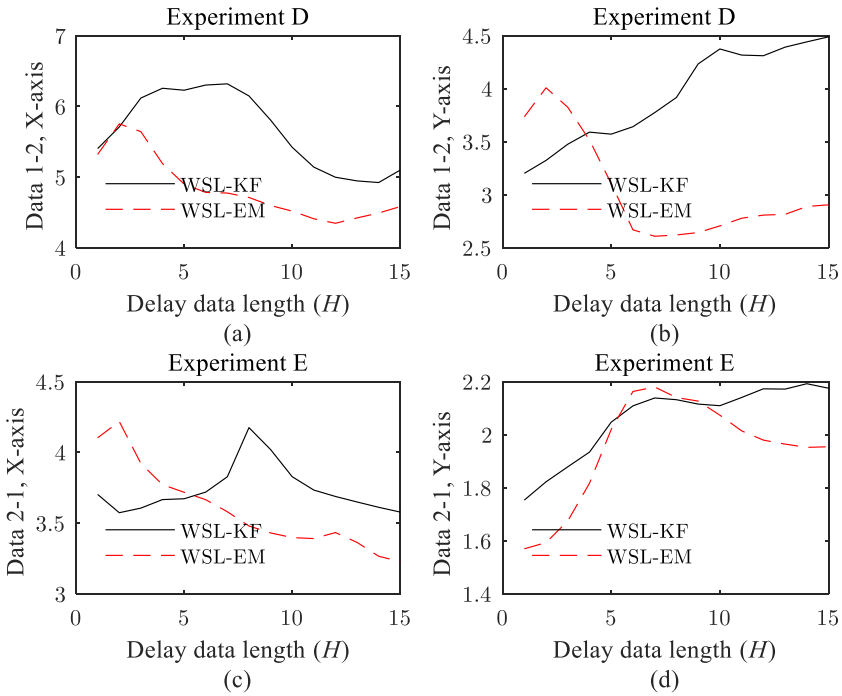

Fig. 7 Estimation error on the X- (left) and Y-axes (right) vs. temporal correlation parameter H in *Experiments* (a, b) D and (c, d) E

that most of the positions decoded are in a reversed state. The instability is only further reduced owing to the introduction of WSL, which cannot fully guarantee a reduction in the decoding error but does guarantee convergence.

In addition, we tested the effects of the number of iteration cycles T of the WSL-KF and WSL-EM algorithms from *Experiments* D and E. As shown in Fig. 9, although in the test results show some differences, the effect is not large. Therefore, it is essential to conform to the T-values corresponding to the X- and Y-axes in the above-mentioned tests. From the comprehensive analysis, it was concluded that the error curve was stable after $T = 3$ in the WSL-KF, so the number of iteration cycles for convergence could be

taken as $T = 3$ at the lowest. In terms of WSL-EM, considering the decoding performance and time cost paid from *Experiments* B and E, the number of iteration cycles for convergence could be taken as $T = 8$ at the lowest, which is the smallest decoding error.

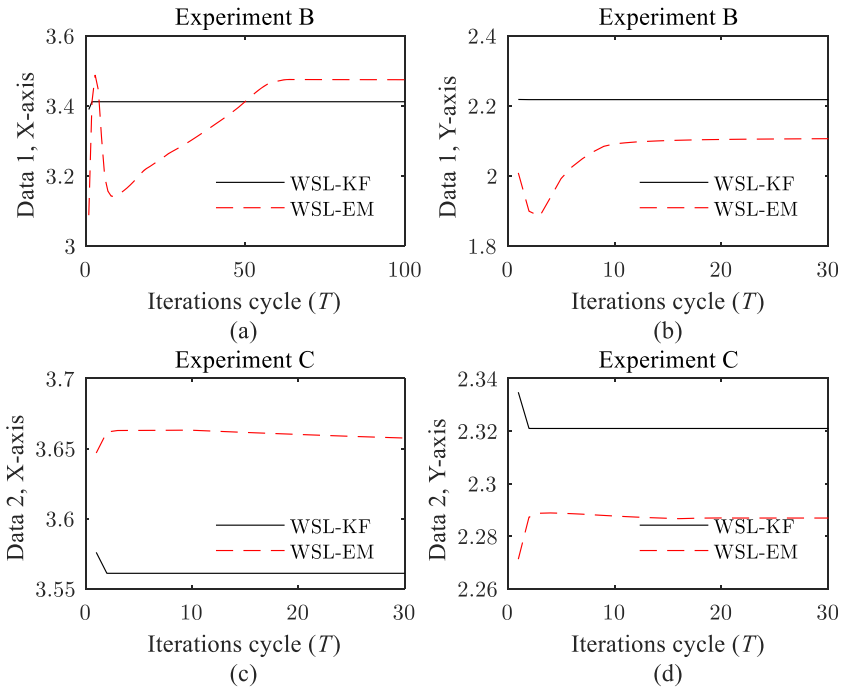

Fig. 8 Estimation error on X- (left) and Y-axes (right) vs. number of iterations cycles T from *Experiments* (**a**, **b**) B and (**c**, **d**) C

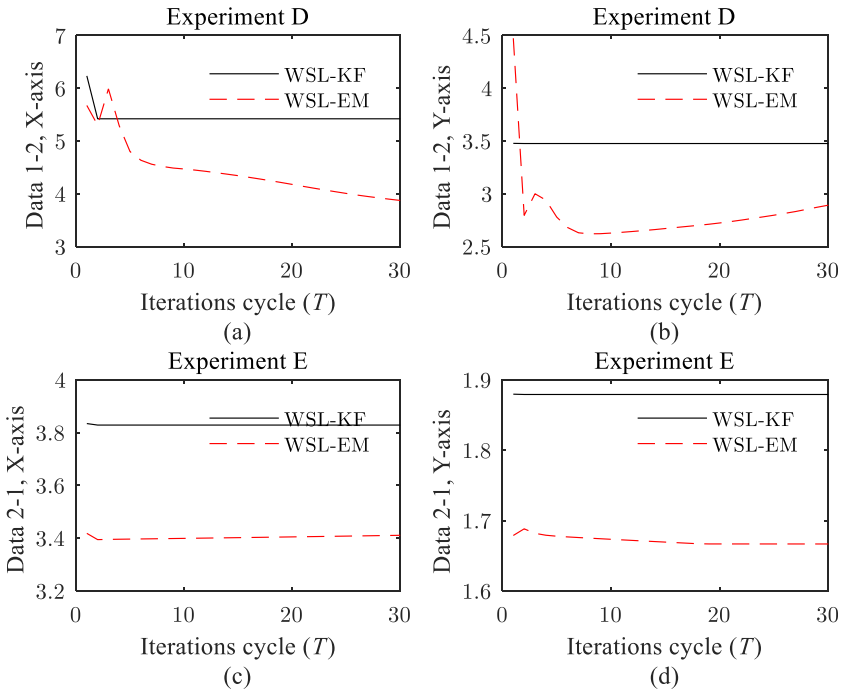

Fig. 9 Estimation error on *X*- (left) and *Y*-axes (right) vs. number of iterations cycles *T* from *Experiments* (**a**, **b**) *D* and (**c**, **d**) *E*

Discussion

In this study, we used WSL to correct the position of the moving finger of a macaque that originally had too large of a decoding error or an inversed error when unsupervised learning was utilized, and the model parameters were further trained for increased accuracy. The WSL-KF and WSL-EM algorithms were used to decode the accurate finger movement position in the test data. The problems encountered in this research, as well as subsequent research directions, are discussed below.

According to the results in Figs. 6 and 7, we believe that the *H*-value affects the decoding performance, which may be related to the similarity of the data characteristics. More significant data similarity increased the decoding accuracy and generalization ability and decreased the *H*-value. Likewise, the reduced similarity between the data decreased the decoding accuracy and generalization ability and increased the *H*-value. In addition, the *H*-value may be related to the characteristics of the algorithm itself. That is, *H* may not be a fixed value, but rather may vary with the degree of similarity between the data or the algorithm itself.

The results according to the numbers of iteration cycles in Figs. 8 and 9 show that the error curve will be in a convergent state with the number of iteration loops and will stabilize in a straight line. However, the error does not necessarily converge to the minimum, but rather may increase. Therefore, if the error is in an increased state before reaching convergence, but the convergence error is within an acceptable range, the iterative cycle number at the state of convergence can be selected. In this manner, the number of iteration cycles for convergence in WSL-KF was found to be $T = 3$, while that for convergence in WSL-EM was determined to be $T = 8$, which is basically at a convergence state.

The experimental results in Tables 1 and 2 demonstrate that the employed WSL model introduces the time correlation *H*-value, which not only has a good decoding effect on the *X*- and *Y*-axes but also causes some decoding errors to be less than or close to the supervised errors. Regarding the decoding error of the two-dimensional plane, the error of the WSL-EM algorithm is less than those of the other approaches, being close to that of supervised learning and smaller than that of unsupervised UCKD.

In this study, the WSL model adopted was divided into two parts, including left and right regions along the *X*-axis and upper and lower regions along the Y-axis, which is equivalent to dividing the XY coordinate plane into four quadrants. According to the four quadrants, the finger movement position decoded by unsupervised learning was judged. The estimated abscissa and ordinate are determined to be positive, with the corresponding positions remaining unchanged, while the position reversed should be changed in Eq. (9). The processed position was used to accurately train the weight parameter. However, unsupervised learning conforms to human self-learning. Completely transforming the WSL into unsupervised learning can be considered; that is, the finger movement position can be completely decoded without label information.

To further study unsupervised learning, it is necessary to determine which features in the neuron spike potential signal

can affect the approximate area of finger positions. That is, the decoding problem of this general region can be directly transformed from regression into classification. At present, deep neural networks can automatically learn and extract the feature values of target objects, generally being applied to target classification. Many classification algorithms in machine learning require further research, such as the SVM algorithm. If higher accuracy can be achieved, then unsupervised learning can as well.

Conclusion

The signals from the brain must be accurately mapped before they can be applied through a neural interface to intuitively control a robotic prosthesis. In this application, external activities can be precisely controlled by the neuron spike signals using the method described in this paper. We also established an experimental basis for the subsequent complete implementation of unsupervised learning. In addition, according to the research results, the workload of data collection can be reduced with knowledge of only a rough area.

In this study, the WSL model was used to decode finger movement positions through neuron spike signals in the motor cortex. The essence of the WSL model is to divide the XY coordinate plane into four quadrants. It is a technical problem of correcting the position decoded by the unsupervised KF and EM, then training the weight parameter of the model. The weight parameter was then used to decode the predicted location. The paper also introduces the time correlation of the neuron spike potential signal and the calculation of the time-dependent H-value in WSL, leading to further reduction of the decoding error of WSL.

Finally, based on the analysis of the five groups of experimental results, the average decoding error yielded by the WSL for the X- and Y-values is less than or similar to that obtained using supervised learning, and less than that obtained by applying unsupervised UCKD and U-EM. Among the XY coordinates, the average error of the WSL-EM is the smallest, only differing by approximately 0.4% from that of the best supervised KF approach. Compared with the unsupervised UCKD algorithm, WSL-EM achieved a performance improvement of approximately 41.3%. Compared with the unsupervised U-EM, WSL-EM exhibited an improvement of approximately 47.7%. These experimental results show that the neural decoding technology based on unsupervised learning, coupled with WSL, can overcome the instability and inaccuracy of unsupervised learning, thereby improving the accuracy of decoding the finger movement position.

Acknowledgements This work was supported in part by the National Natural Science Foundation of China (No. 61762093), the 17th batches of Young and Middle-aged Leaders in Academic and Technical Reserved Talents Project of Yunnan Province (No. 2014HB019), the Key Applied and Basic Research Foundation of Yunnan Province (No. 2018FA036), the Program for Innovative Research Team (in Science and Technology) in University of Yunnan Province, and the Science Research Fund Program in Education Department of Yunnan Province (No. 2018Y106).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval All applicable international, national, and/or institutional guidelines for the care and use of animals were followed.

Informed consent This work is informed consent by any of the authors.

References

1. Kim KT, Suk HI, Lee SW. Commanding a brain-controlled wheelchair using steady-state somatosensory evoked potentials. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2018;26:654–65.
2. Caporusso N. Issues, challenges and practices in advancing pervasive human-computer interaction for people with combined hearing and vision impairments. Ph.D. dissertation, Computer Science and Engineering, IMT Institute for Advanced Studies, Lucca, Italy; 2012.
3. Jacobo FV, Chu LY, Kahori K, Yu W. 3D continuous hand motion reconstruction from dual EEG and EMG recordings. In: Proceedings of International Conference of Intelligent Informatics and Biomedical Sciences (ICIIBMS); Okinawa, Japan, 2015. p. 28-30.
4. Fan JJ, Tian F, Du Y, Liu ZJ, Dai GZ. Thoughts on human-computer interaction in the age of artificial intelligence. Scientia Sinica Informationis. 2018;48:361–75.
5. Tian L, Zimmerman B, Akhtar A, et al. Publisher Correction: Large-area MRI-compatible epidermal electronic interfaces for prosthetic control and cognitive monitoring. Nature Biomedical Engineering. 2019;3(3):194–205. https://doi.org/10.1038/s41551-019-0347-x.
6. Borra E, Gerbella M, Rozzi S, Luppino G. The macaque lateral grasping network: a neural substrate for generating purposeful hand actions. Neuroscience and Biobehavioral Reviews. 2017;75:65–90.
7. Philippsen A, Nagai Y. A predictive coding model of representational drawing in human children and chimpanzees. In: Proceedings of 2019 Joint IEEE 9th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob), Oslo, Norway, 2019. p. 171-176.
8. Georgopoulos AP, Lurito JT, Petrides M, Schwartz AB, Massey JT. Mental rotation of the neuronal population vector. Science. 1989;243:234–6.
9. Chen Z, Takahashi K. Sparse Bayesian inference methods for decoding 3D reach and grasp kinematics and joint angles with primary motor cortical ensembles. In: Proceedings of Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE, Osaka, Japan, 2013. p. 3-7.
10. Haykin S. Neural networks and learning machines. 3rd ed. Prentice Hall: Upper Saddle River, NJ; 2008.
11. Czanner G, Eden UT, Wirth S, Yanike M, Suzuki WA, Brown EN. Analysis of between-trial and within-trial neural spiking dynamics. Journal of Neurophysiology. 2018;99:2672–93.

12. Aghagolzadeh M, Truccolo W. Latent state-space models for neural decoding. In: Proceedings of 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Chicago, USA. 2014. P. 3033-3036.
13. Rule ME, Sanguinetti G. Autoregressive point-processes as latent state-space models: a moment-closure approach to fluctuations and autocorrelations. Neural Computation. 2018;30:2757–80.
14. Syed AM, Gunasekaran N, Esther RM. Robust stability of hopfield delayed neural networks via an augmented L-K functional. Neurocomputing. 2017;234:198–204.
15. Paranjape PN, Dhabu MM, Deshpande PS, Kekre AM. Cross-correlation aided ensemble of classifiers for BCI oriented EEG study. IEEE Access. 2019;7:11985–96.
16. Li SK, Jiang YB. Semi-supervised sentiment classification based on sentiment feature clustering. Journal of Computer Research and Development. 2013;50:2570–7.
17. Xue ML, Wu HF, Zeng Y. Unsupervised CKF decoding for macaque motor cortical spikes. Acta Automatica Sinica. 2017;43:302–12.
18. Zhou ZH. A brief introduction to weakly supervised learning. National Science Review. 2018;5:44–53.
19. Zhang XD. Modern signal processing. 3rd ed. Beijing, China: Tsinghua University Press; 2015.
20. Wu HF, Feng JY, Zeng Y. Neural decoding for macaque's finger position convolutional space model. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2019;27:543–51.
21. Smith AC, Frank LM, Wirth S, Yanike M. Dynamic analysis of learning in behavioral experiments. Journal of Neuroscience. 2004;24:447–61.
22. Wallisch P, Lusignan ME, Benayoun MD, Baker TL, Dickey AS, Hatsopoulos NG. Matlab for neuroscientists. 2nd ed. London: Academic Press; 2014.
23. Raschka S. Python machine learning. 2nd ed. Birmingham, UK: Packt Publishing; 2015.