Master's Thesis

Decoding arm kinematic parameters from motor cortical ensemble activity using long short-term memory

Jisung Park

Department of Human Factors Engineering

Graduate School of UNIST

2018

Decoding arm kinematic parameters from motor cortical ensemble activity using long short-term memory

Jisung Park

Department of Human Factors Engineering

Graduate School of UNIST

Decoding arm kinematic parameters from motor
cortical ensemble activity using long short-term
memory

A thesis/dissertation
submitted to the Graduate School of UNIST
in partial fulfillment of the
requirements for the degree of
Master of Science

Jisung Park
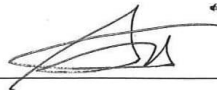
07/02/2018 of submission

Approved by
_____
Advisor
Sung-Phil Kim

Decoding arm kinematic parameters from motor cortical ensemble activity using long short-term memory

Jisung Park

This certifies that the thesis/dissertation of Jisung Park is approved.

07/02/2018 of submission

_____

Advisor: Prof. Sung-Phil Kim

_____

Prof. Oh-Sang Kwon

_____

Prof. Dongil Chung

ABSTRACT

Brain machine interface (BMI) is the interface which converts the neural signal recorded from the subject into the intention such as arm movement or grasping. One of the key topic in the BMI research area is to decode the neural signal from the motor cortex in the brain and extract information related with the movement such as velocity, position and speed of arm. By interpreting the signal, the tetraplegia can get a chance to overcome the obstacle by controlling the robot arm according to their intention interacting with the environment.

To achieve these goal, several algorithms such as Kalman Filter or optimal linear estimation have been used. Further, modern machine learning algorithms such as Long Short-Term Memory(LSTM) However, most of the decoders have focused on the decoding of the velocity parameter even though there were evidences that the neural activity of the primary motor cortex encodes the directional information and speed information differently.

The developed decoder in this research reconstructed the velocity of an arm movement with two separate LSTM decoder, which is designed for individual decoding of speed and direction. Because there is neural evidence that the direction is encoded nonlinearly in the motor cortex and the speed variable has nonlinear characteristics, the nonlinear prediction algorithm, which is LSTM, was used as the predictor. Also, velocity Kalman filter (VKF) and velocity LSTM (VLSTM) were compared with the decoder. The performance was measured for the data provided from the Collaborative Research in Computational Neuroscience - Data sharing (CRCNS). The task was center – out reaching movement and the primary motor cortex signal of macaque monkey was recorded. Specifically, the correctness of the velocity reconstruction was tested with following measurements: angular difference, correlation coefficient, mean absolute error. Further, the correctness of the position reconstruction was tested with following measurements: Euclidean distance with true trajectory. Also, the effectiveness of the reconstructed trajectory was measured with following indexes: Euclidean distance with straight line from home to target, hit rate, distance to target according to the movement time, movement directional change (MDC), orthogonal directional change (ODC).

The results in aspects of how accurately the decoder predicted the kinematics showed that the new decoder predicts the movement direction accurately than the other two decoders. The quality of position reconstruction of the SDLSTM was significantly better than the velocity Kalman filter and was similar with the VLSM. Also, the SDLSTM's trajectory could be acquired the target in highest frequency. Also, the model complexity of the speed and direction predictor in the SDLSTM were different with each other. It implies that the encoding strategy of the two variables in the motor cortex can be different. Finally, we could identify that the SDLSTM increased the overall decoding performance.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

X

1. Introduction

1.1 Introduction to Brain Machine Interface

Brain Machine Interface (BMI) indicates devices that decode intention of organism such as monkey or human from brain signal and transfer the decoded intention into other useful control signal such as motor control signal. For example, monkey can take water using robot hand controlled with brain signal. It can be achieved from intracortical signal or extracortical signal. Extracortical signal, such as Electroencephalogram(EEG), could be acquired without surgery. It has characteristic that has low temporal resolution, high noise level. In the other hand, the intracortical signal that records the signal from the population of neurons directly has small amount of noise and high temporal resolution although it records only very small cortex area.

The research of BMI using intracortical signal have been progressed with the neural and behavioral data recorded in the macaque monkey. In general, the center-out task is given to the monkey.
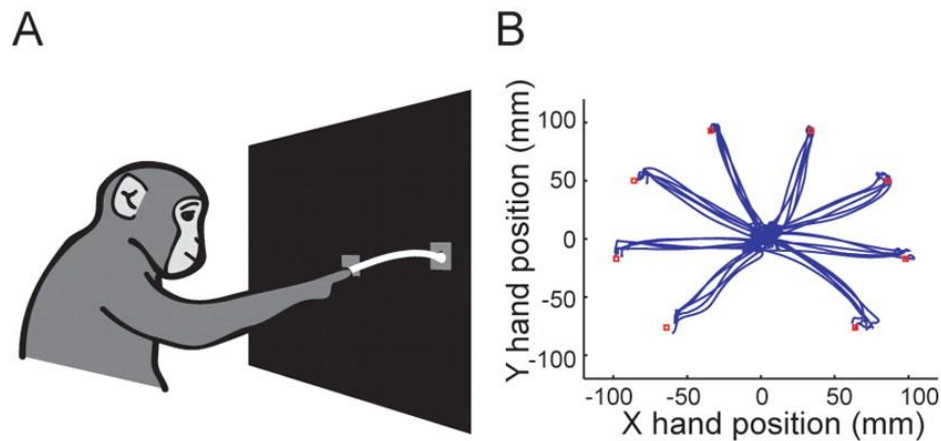
Figure 1 Center out task. (A) a monkey moves the endpoint of finger from home position which is at the center to the target position which is outer position (B) Trajectory of the task. The target positions were marked as the red square. (Cynthia A. Chestek, 2007)

In the task, the monkey places his own hand to the home position in a specific duration. Then, a specific outer target which is placed in radial around the center with fixed radius is given on the screen. The monkey performing the task moves the hand to the outer target and if the hands are placed around the target in a duration, the trial is regarded as the success trial. The targets are distributed with the fixed interval of angle (Figure 1).

During the monkey perform the center-out task, the neural signal from the cortex of the monkey is recorded. Matching the recorded signal with the behavioral data make it possible to extract the kinematic information such as arm movement from the neural signal. Especially, the neural signal in the motor cortex is known as encoding the information related with the arm movement direction. According to the movement direction, the firing rate which is the number of action potential of a neuron

in a unit time change based on the cosine relation with the movement direction (Bagrat Amirikian *et al.*, 2000) The Figure 2 indicates the encoding example increase or decrease according to the movement direction. Each black dot shows the mean firing rate in the direction and the error bar indicates the standard deviation of the firing. This tuning property is called as the cosine tuning and the movement direction shows the maximum firing rate is the preferred direction.
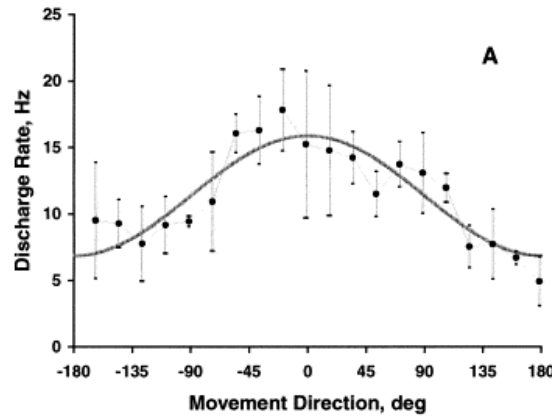
Figure 2 The directional information encoding of the motor cortex neuron (Bagrat Amirikian, 2000).

1.2 Kinematic information encoding in the motor cortical activity

Motor cortical ensemble activity encodes kinematic parameters of arm movement such as velocity, speed and direction with the number of action potential in a unit time, which is called as firing rate. Especially, there are neurons encoding the arm movement direction with firing rates in the primary motor cortex. These neurons' firing has relation with the movement direction, which is expressed with the cosine function (Georgopoulos AP *et al.*, 1982). The $\theta_{preferred}$ is the preferred direction that has peak firing rate and the fr indicates the mean firing rate during the movement.

$$\text{fr} = \text{b} + \text{k} \cos(\theta - \theta_{preferred}) \tag{1}$$

Bagrat's work showed that the directional information was encoded nonlinearly in the cortex with model similar with the von Mises distribution (Bagrat Amirikian, 2000).

$$fr = a_0 + \exp(a_1 \cos(\theta - \theta_{preferred})) \tag{2}$$

The results showed that the nonlinear tuning model explained better than standard cosine tuning model.

There is research shows that the directional information is encoded stronger than the speed information in the motor cortex (Golub, M. D. et al., 2013). In the research, they measured the mutual information between the single neuron's firing activity in the motor cortex and the kinematic variables, direction

4

and speed in 50ms window shifting the window on the task time. They found when the directional information and the speed information becomes maximum on the movement. The Figure 3.A shows that the maximum directional information(MDI) and maximum speed information(MSI) of each recorded neuron. When the MDI was larger, it was marked as blue. For the opposite case, it was marked as the red color. Black indicates there were no significant difference. The result shows that there exist more number of the motor cortical neurons that encodes the direction stronger than the speed. The authors suggested several interpretations. One of them is that the speed information is encoded robustly in the motor cortex and the analysis in the research didn't catch the robust information.
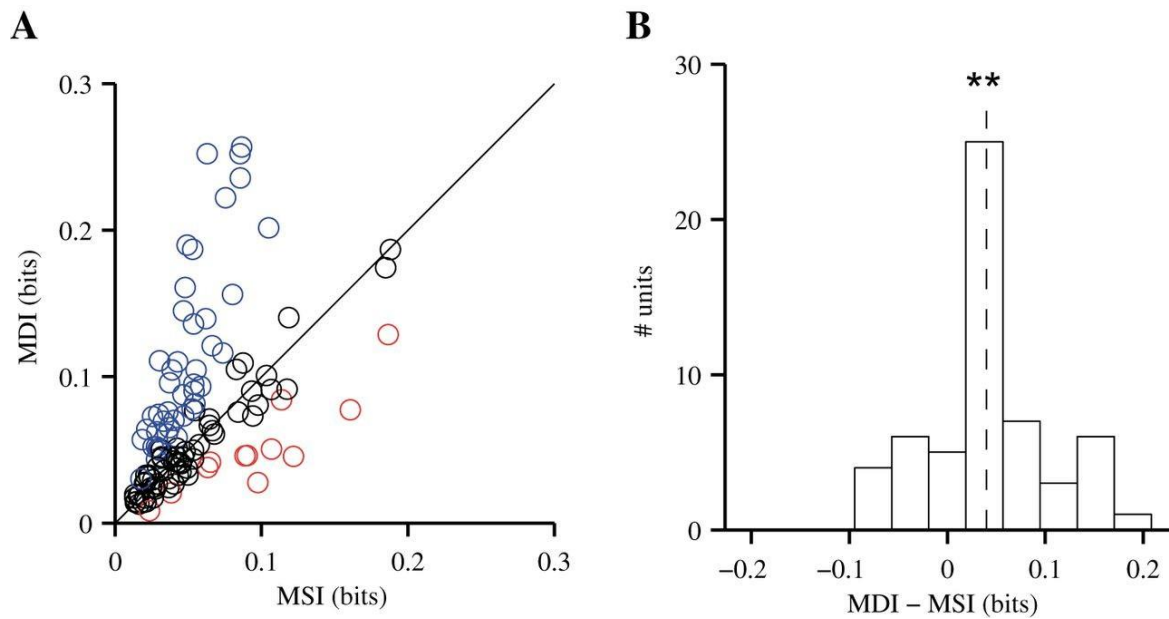
Figure 3 (Left) Motor cortical neuron's maximal directional information (MDI) and the maximal speed information (MSI) comparison. (Right) The histogram of the number of cells according to the difference between MDI and MSI (Golub, M. D. *et al.*, 2013)

Also, the Zheng Li's research showed that the velocity information is encoded nonlinearly in the motor cortex ensemble activity. They claimed the nonlinear relation with the use of the unscented Kalman filter (UKF) as the neural decoder.

The Kalman Filter is the linear prediction algorithm. The state vector at time t has linear relation with the state vector at time t+1. The encoding model representing the relation between the state vector and neural activity has also linear relation. However, Zheng Li showed the nonlinear quadratic model could capture the neural activity pattern according to the velocity and position of arm movement better than the linear model (see Figure 4) and based on this finding, they used the unscented Kalman filter (UKF), which used nonlinear encoding model, as the neural decoder (Z Li *et al.,* 2009).
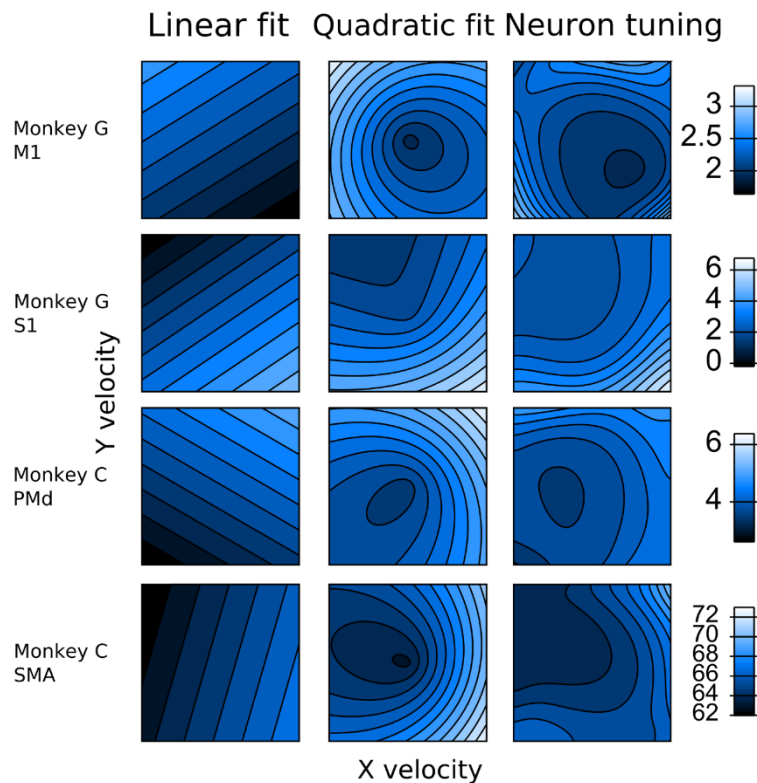
Figure 4 Comparison of linear and quadratic tuning models of movement velocity (Z Li *et al.,* 2009)

1.3 Nonlinearity of the speed variable

The speed variable has nonlinear characteristic in the center-out reaching task. The figure 5 shows that the speed profile in the trials of the reaching task which was performed the Northwestern university (R. D. Flint *et al.*, 2012). Each color represents the target direction. The speed increases or decreases drastically in the movement. In other words, the arm movement speed has nonlinear characteristic in the center out reaching task.
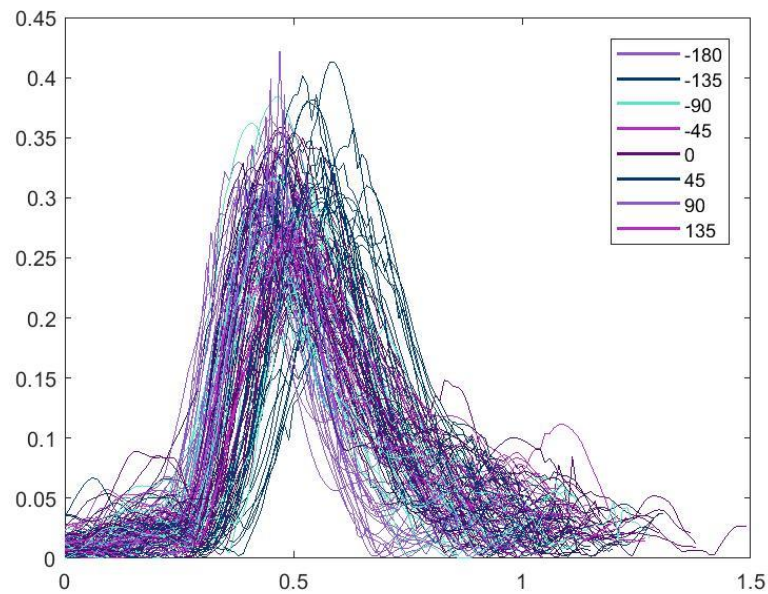
Figure 5 Speed profile in the center-out reaching task. Each profile is that of each trial and the color represents the target direction

1.4 Neural decoder

1.4.1 Kalman Filter

The Kalman filter is an algorithm that predict the state variable, such as position, velocity, acceleration, with prediction step and correction step. The prediction step predicts the state of next time with current state with linear equation. The correction step corrects the predicted state using measured state. How much the measured state will be reflected to the predicted state is determined with the Kalman gain. As the neural decoder, the Kalman filter can be expressed with following equations:

$$Z_k = H_k X_k + q_k \tag{3}$$

$$X_{k+1} = A_k X_k + w_k \tag{4}$$

The Z_k means that the mean firing rate of N number of neurons in a specific time window, thus it has N dimensional vector. The X_k is state variable, such as position and velocity of arm endpoint. The q_k is noise from normal distribution of zero mean. The distribution has covariance matrix as Q_k. The w_k is noise from the normal distribution of zero mean, which has covariance matrix W_k. The matrix A represents the relation between kinematic variables of time k and time k-1. The matrix H represents how the kinematic variables were encoded into the neural activity. The A, H, W, Q can be estimated with least mean square solution.

The prediction step is processed with following process. First, the priori, $\hat{X}_k^-$, is acquired with the transition matrix A and error covariance matrix of priori, P_k^-, is computed.

$$\hat{X}_k^- = A\hat{X}_{k-1} \tag{5}$$

$$P_k^- = AP_{k-1}A^T + W \tag{6}$$

In the correction step, the priori is corrected taking into the neural response. Also, the posterior error covariance matrix, P_k, is updated. The K_k is the Kalman gain matrix. The full explanation of the Kalman filter was introduced in the works of Wu, Wei (Wu, Wei *et al.*, 2003)

$$\hat{X}_k = \hat{X}_k^- + K_k(z_k - H\hat{X}_k^-) \tag{7}$$

$$P_k = (I - K_k H)P_k^- \tag{8}$$

$$K_k = P_k^- H^T (HP_k^- H^T + Q)^{-1} \tag{9}$$

The important assumption of the Kalman filter is that the kinematic variable is encoded into the cortical neural activity of motor cortex with a linear relation. However, as shown in the previous literature survey,

the relation between neural activity and the kinematic variable is not linear. Thus, to improve the decoding performance, the nonlinear prediction algorithm, such as unscented Kalman filter and feedforward neural network, have been tried.

1.4.2 Feedforward Neural Network (FNN)

The Feedforward neural network is the prediction algorithm which is consisted with input layer, hidden layer, output layer, and activation function. The input data is linearly summed with the parameter 'weight' and it is delivered into the activation function. The following diagram explain the process:
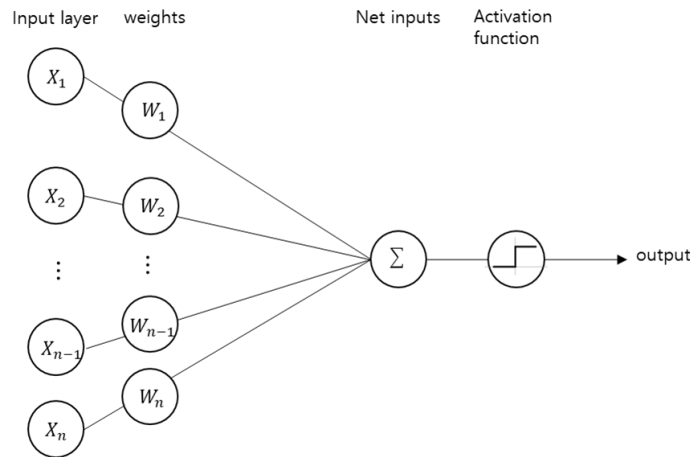
Figure 6 Neural network input processing

The activation function and net inputs are called as the hidden node. The layer which is consisted only with the hidden node is called as the hidden layer. After concatenating several hidden layers, the output layer is added after the last hidden layer. The output layer represents the variable predicted. The network structure is called as the Feedforward Neural Network. The following diagram explain the process:
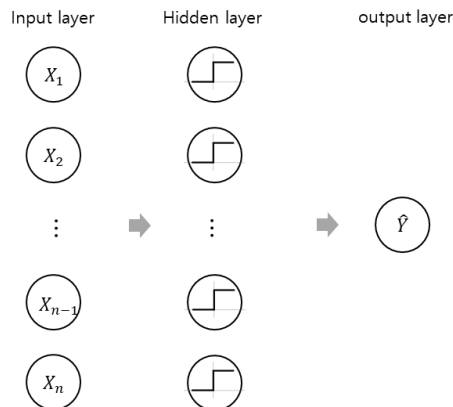
Figure 7 Structure of the feedforward neural network

The fully connected (FC) layer is the structure that the two layers are concatenated. Also, the error trying to minimize with the network is called as the loss function. In general, the Mean Squared Error (MSE) is used as the loss function in the regression problem:

$$E = \frac{1}{n}\sum_{i}(y_i - \hat{y}_i)^2 \tag{10}$$

The n is the number of samples. The y_i is the i-th observation and the $\hat{y}_i$ is the i-th prediction. The weight w is updated to minimize the loss function to make the accuracy of prediction better. It can be given with following equation.

$$W_t = W_{t-1} - \varepsilon\frac{dE}{dw} \tag{11}$$

The error can be expressed with weights and the negative gradient with respect to weights represents the direction that decrease the error. It is weight updating strategy called as the gradient descent.

1.4.3 Long Short-Term Memory (LSTM)

Long Short-Term Memory was firstly developed by Sepp Hochreiter and Jurgen Schmidhuber in 1997 (Sepp Hochereiter *et al.*, 1997) It has advantages that it can extract information in the time-structured data. It was designed for handling the 'long-term dependency' problem that the transferred information from the past became week when the timing between input and predicted output became long

Basically, the LSTM is consisted with a module that has cell state, explicitly saving the information from the past input, and the hidden state, representing the information inferenced from the current input and the cell state of the previous time step. The module can be expressed with following diagram:
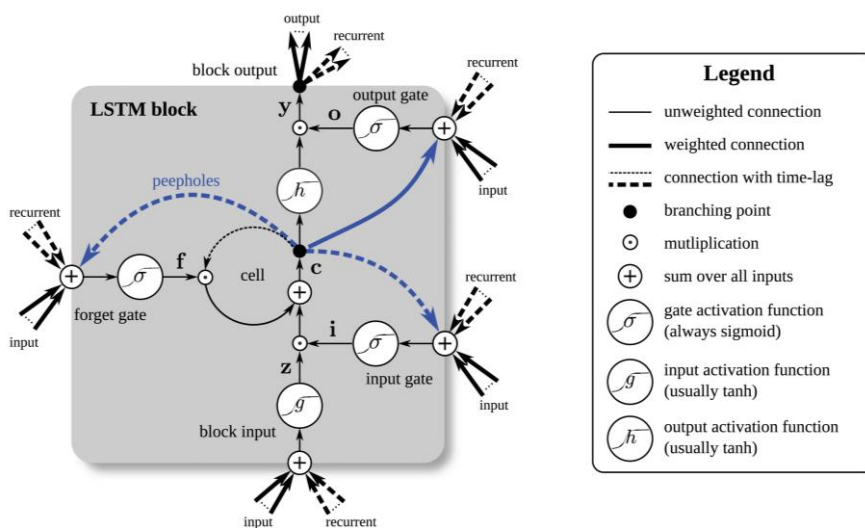
Figure 8 LSTM block (K. Greff *et al.*, 2015).

The C represents the cell state, the h represents the hidden state at time t and the f is the output of the 'forget gate'. The 'forget gate' decides how much information from the cell state at time t-1 will be passed to the cell state at time t. The i is the output of the 'input gate'. The 'input gate' decides how much information from the newly estimated cell state is saved to the cell state at time t. The cell state at time t is estimated with the linear sum of previous cell state and newly estimated cell state. Finally, the o_t is the output of the 'output gate'. It decides how much information from the cell state at time t become hidden state at time t. The h_t indicates the hidden state at time t. Further, the estimated cell state and hidden state at time t is passed to the next module that handling the input data at time t+1. Also, the h_t is given to the fully connected layer, which is combination of the input layer and another layer. In here, the h_t becomes the input layer and it is transferred to the nonlinear output layer. This process can be represented with following diagram:
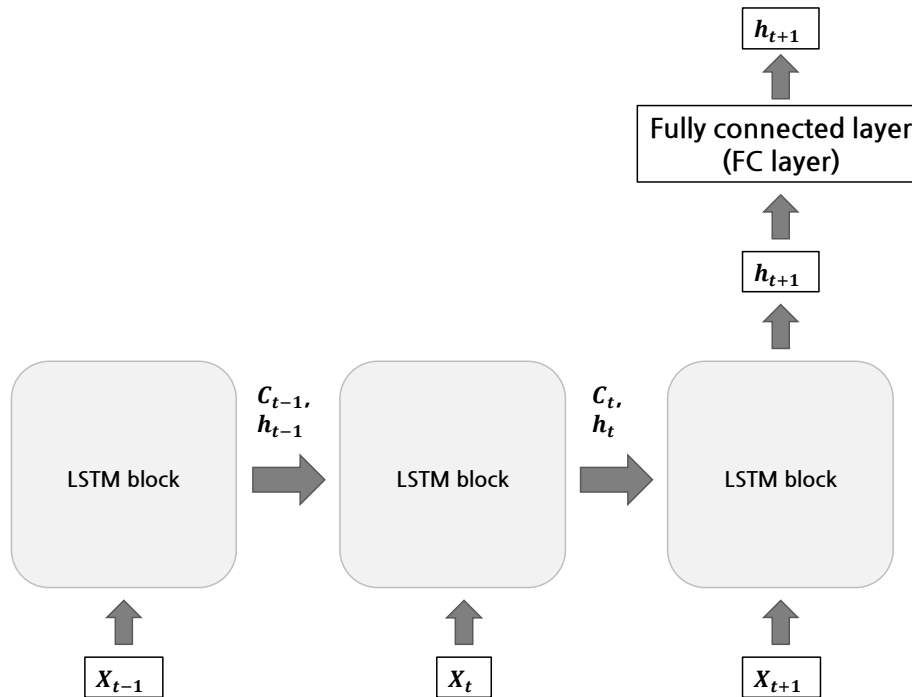
Figure 9 LSTM block sequence. Input vector X at each time step is given to the LSTM cell

1.5 Parameter update algorithm

To update the weights of a network, the gradient descent algorithm is used, which updates the weights in a direction of decreasing the error of prediction, which is expressed with following equation:

$$W_t = W_{t-1} - \varepsilon \nabla_\theta f_t(\theta_{t-1}) \tag{12}$$

The stochastic gradient descent (SGD) is the strategy that calculating the $\nabla_\theta f_t(\theta_{t-1})$,which is gradient of the loss function $f(\theta)$ with respect to the weights θ on the parts of training sample instead of calculating on full training set. It has advantage that can avoid local minima of the loss function.

There are lots of variations of the SGD. First one is the *momentum method*. It is the updating algorithm using the momentum of pervious updates.
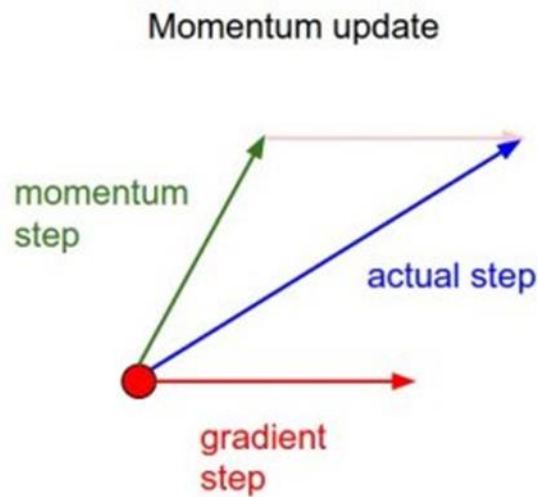
Figure 10 Moment updates (CS231n)

In the figure, the gradient step is the updating direction of weights acquired from current training batch. The momentum step is the averaged previous updating directions of weights. The final updating direction becomes the blue arrow. With the momentum algorithm, the networks can reach to the optimal performance faster.

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta f_t(\theta_{t-1}) \tag{13}$$

$$\theta_{t+1} = \theta_t - v_t \tag{14}$$

The equations explain the momentum algorithm that it accumulates the previous gradients to the v_t affected by the momentum step to the current weights θ_t. The algorithm can go to the optimal loss faster with less inefficient weight update (David E. Rumelhart et al., 1986).

Another variant is the Adam (Adaptive Momentum Estimation) algorithm that combines the RMSProp and the momentum algorithm (Diederik P. Kingma *et al.*, 2014). The algorithm is expressed with

following equation:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \nabla_\theta f_t(\theta_{t-1})$$

(15)

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)(\nabla_\theta f_t(\theta_{t-1}))^2 \qquad (16)$$

The m_t and v_t accumulates the previous weights updates in similar way of the momentum algorithm.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \qquad (17)$$

$$\hat{v}_t = \frac{\hat{v}_t}{1 - \beta_2^t} \qquad (18)$$

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} \hat{m}_t \qquad (19)$$

The $\hat{m}_t$ and $\hat{v}_t$ are acquired from the m_t and v_t to reduce the bias of initial updates. The $\hat{m}_t$, which is accumulated gradient, is scaled by $\hat{v}_t$ to update the weight θ_t. The scaling strategy is similar with that of the RMSProp algorithm (G hinton *et al.*, 2012). In the RMSProp algorithm, the size of squared previous gradient is used to scale the updates without the momentum equation (18). We used the Adam algorithm to update the velocity LSTM and the SDLSTM

1.6 Parameter searching algorithm

The nonlinear prediction algorithm introduced in the previous section has different performance according to the set of parameters, such as the number of hidden nodes, the number of epochs, batch size and learning rate. Thus, to find the optimal parameter is important issue. There are methods to search the optimal parameter. The basic method is the grid search. The strategy is to find the optimal parameter from the all of combination in the specified hyperparameter subset. Another strategy is the random search that tests the parameter combination randomly sampled from the specified distribution, such as uniform distribution. The James Bergstra and Yoshua Bengio argued that the random search is better than the grid search in empirical and theoretical way. Because not all the parameter does not have same amount of importance on the performance of the network, when the parameters are tested in equally spaced grid, the parameter combination of high performance could be missed. The following

diagram shows the relation (James Bergstra *et al.*, 2012).

Grid Layout

Random Layout

Unimportant parameter

Unimportant parameter
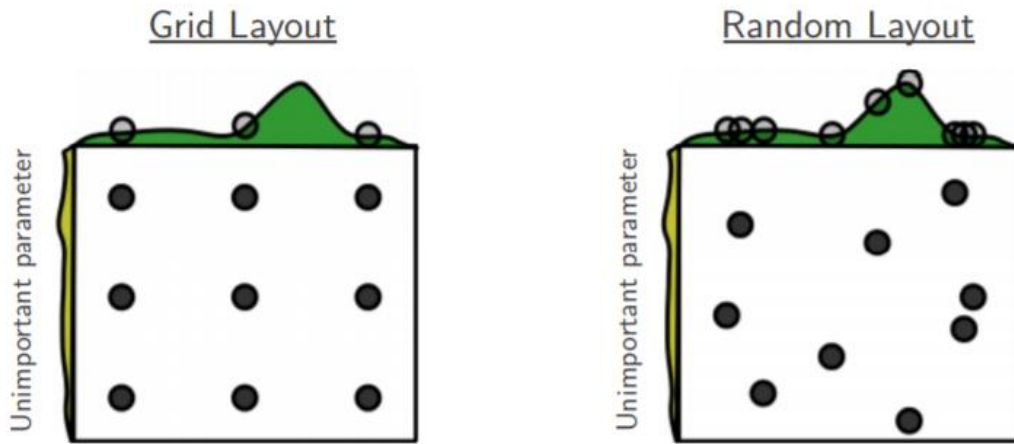
Figure 11 Concept diagram of grid search and random search (James Bergstra *et al.*, 2012)

In the figure 11, the white square represents the subspace of two parameters, x and y. The green and yellow plot represents the effectiveness of x and y for optimizing the network. Because the green plot has more larger value than the yellow plot, the parameter x is regarded as the important parameter. The grid search strategy can miss the optimal parameter set. In the other hand, the random search has more chance to get better set of parameters by unevenly searching the parameter space. In our work, the random search strategy was used to optimize the Velocity LSTM and the Speed-Direction LSTM

1.7 Performance comparison of the linear and nonlinear decoder

In the previous section, the neurobiological grounds for separating the direction and speed parameter was introduced. Also, several neural decoding algorithms for predicting the kinematic parameters was reviewed. Further, the parameter updating and searching algorithms are introduced. In this section, the performance of linear and nonlinear decoder is reviewed.

The nonlinear neural decoder has better decoding performance in general than the linear one. It implies that the kinematic parameters are nonlinearly encoded in the motor cortical ensemble activity. In this section, some researches that used nonlinear prediction algorithm as neural decoder and improved the decoding performance are reviewed although the velocity parameter was directly decoded instead of explicitly decoding the direction and speed. A work was that of the Joshual I. Glaser (Joshual I. Glaser, 2017). In this paper, lots of decoder such as Wiener Filter, Wiener Cascade, Kalman Filter, Support Vector Regression, XGBoost, Feedforward Neural Net, Recurrent Neural Net, GRU, LSTM, Ensemble model was tested on the decoding of kinematic parameters from the motor cortex and somatosensory cortex of a monkey and from the hippocampus of a rat. The Weiner Filter, Wiener Cascade, Kalman

Filter is linear decoding algorithm. In the other hand, the other decoders are nonlinear. The results showed that the nonlinear decoding algorithm has better performance than the linear (Figure 12).
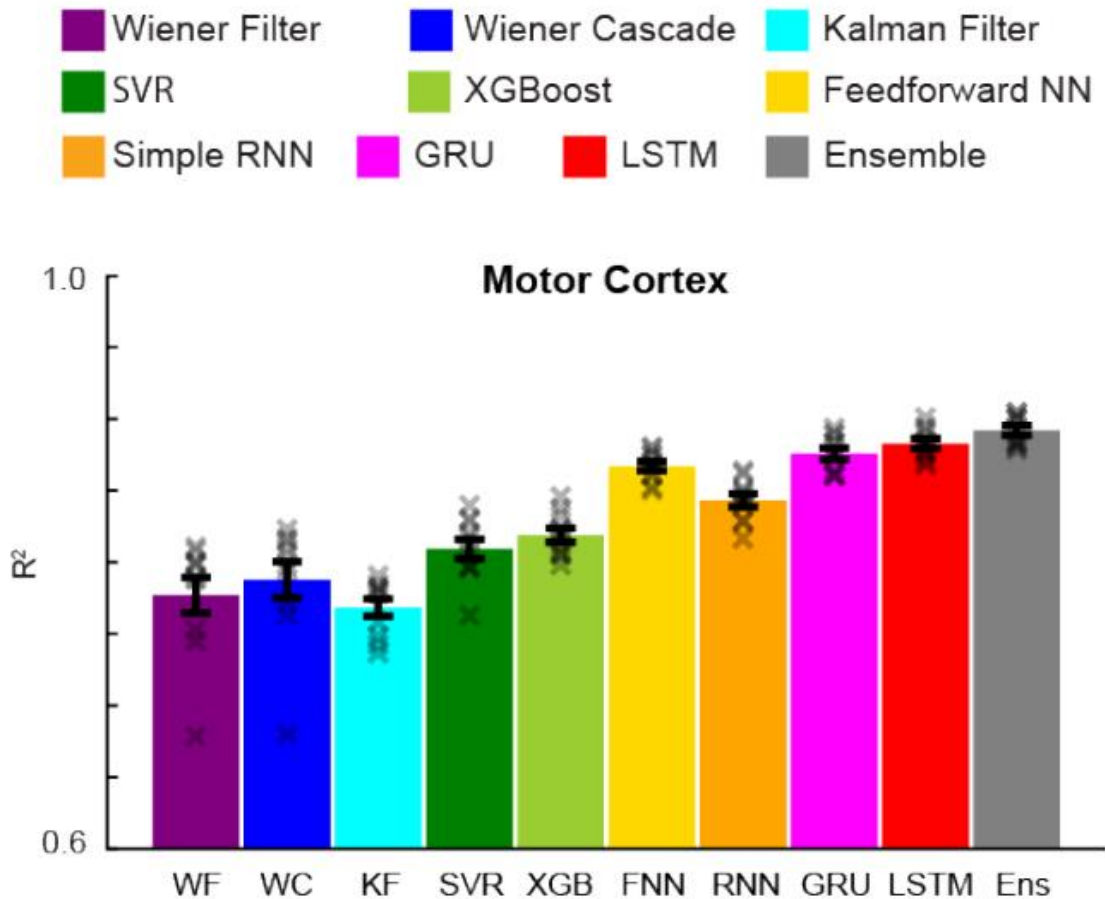
Figure 12 Velocity decoding from the neuronal activity of the motor cortex (Joshual I. Glaser, 2017)

Another work is that comparing the linear and nonlinear decoding algorithm for motor cortical activity decoding to predict the arm movement trajectory (Kai Xu et al., 2011). It compared the linear algorithm, Kalman filter, with the nonlinear algorithm, general regression neural network (GRNN) and support vector regression (SVR). The results showed that the nonlinear algorithm performed better than the linear one. The GRNN and SVR showed lower RMSE than that of Kalman filter (Figure 13)
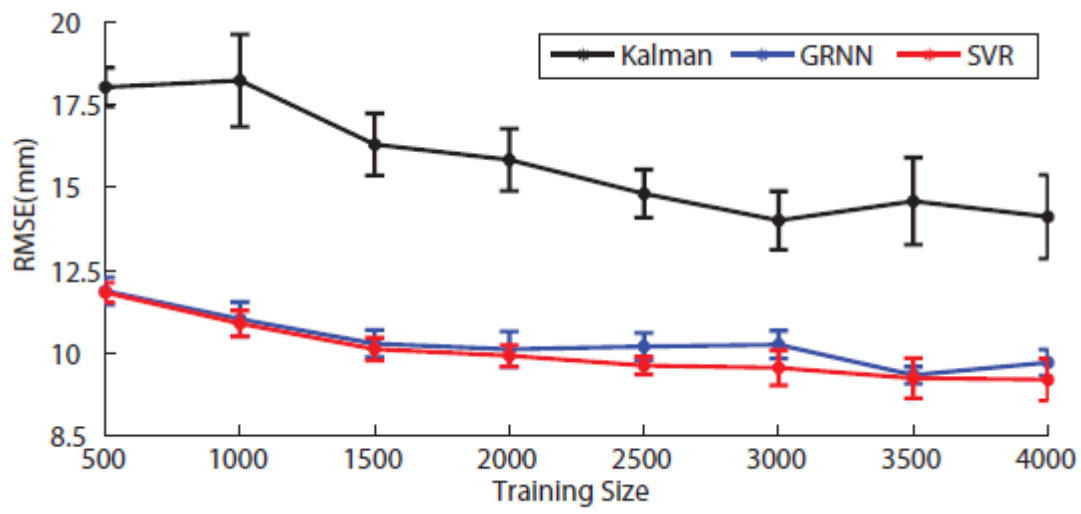
Figure 13 Comparison between linear and nonlinear decoding algorithm (Kai Xu et al., 2011)

1.8 Research aim

As introduced in the section 1.1, the BMI has goal to achieve overcoming the physical disability such as tetraplegia by transferring the brain signal into the intended movement of the robot arm. Many neural decoders have been developed to improve the accuracy of the kinematic variable prediction of the arm movement starting from the Kalman filter to the LSTM (section 1.4 and section 1.7). In this research, we tried to improve the decoding performance by developing the new neural decoder, which is called as the speed-direction LSTM.

In the section 1.2, we introduced a research showing the directional information is strongly encoded in the motor cortical activity than the speed information is encoded. It implies that the speed information is encoded in more robust way that the analysis in the research could not capture. These results give the evidence that the speed and direction could be encoded differently in the aspects of time scale or the encoding method such as spike timing and firing rate. It requires the independent decoding methods for each variable to make each algorithm fit to each encoding method of those variables. Thus, we tried to decode the neural activity with separate decoding model into the direction and speed variable.

In the section 1.1, we introduced the center-out reaching task and the characteristics of the ensemble neurons in the motor cortex that is called as the cosine tuning property. The property modulates the firing activity of the motor cortex neuron according to the movement direction. It implies that the directional information can be explained with the linear equation of directional component of arm movement (Georgopoulos AP *et al.*, 1982). Further, another research showed that the direction can be explained better with nonlinear model (section 1.2). Thus, we tried to predict the direction variable with nonlinear algorithm. Also, the speed variable was predicted with the nonlinear algorithm because the variable showed nonlinear characteristic in the reaching task (section 1.3).

In the section 1.7, we introduced a research that compares the decoding performance of the decoders previously applied in the arm movement prediction, especially velocity parameter. The figure 12 and 13 shows that the nonlinear prediction algorithm was better than the linear prediction algorithm. Among the nonlinear neural decoder, LSTM showed best performance except the Ensemble method. Thus, we developed the neural decoder, speed-direction LSTM, that predicts speed and direction variable separately with the LSTM algorithm. We didn't consider the ensemble method as the neural decoder for predicting the variables because the ensemble method is the coupled decoder combining the FNN with the other decoders except that the KF and thus, the LSTM predictor is simpler model than the ensemble method. The goal of this research was to identify whether the proposed decoder improve the decoding performance when it is compared with that of neural decoders previously suggested. The decoding performance can be measured in two aspects, accuracy and effectiveness. In this research, we regarded the aspects of accuracy as more important criteria.

THIS PAGE INTENTIONALLY LEFT BLANK

2. Methods

THIS PAGE INTENTIONALLY LEFT BLANK

2.1 Data description

The open data provided from the Collaborative Research in Computational Neuroscience –Data sharing (CRCNS) was used to the analysis. The data was recorded in the Northwestern University.

Two rhesus monkeys participated to the experiments, which was center-out task and random-target task. We only analyzed the data recorded in the center-out task. In the eight-target center-out task, monkey grasped and moved the two-link manipulandum to control the cursor. The eight targets were placed around a circle with radius of 10 cm with 45 degree interval. The home position at which the monkey starts each trial was placed at the center of the circle. In each trial, the monkey holds the home position with a duration randomly given between 0.5 ~ 0.6s. Then, the randomly selected outer target, which was 2cm square, was brighten and the center position became darken. The monkey had to reach to the outer target and hold for random duration between 0.2 s and 0.4 s for the success of the trial. The eight targets were given equally in random order. The data of first experiment recorded from the monkey C was analyzed.
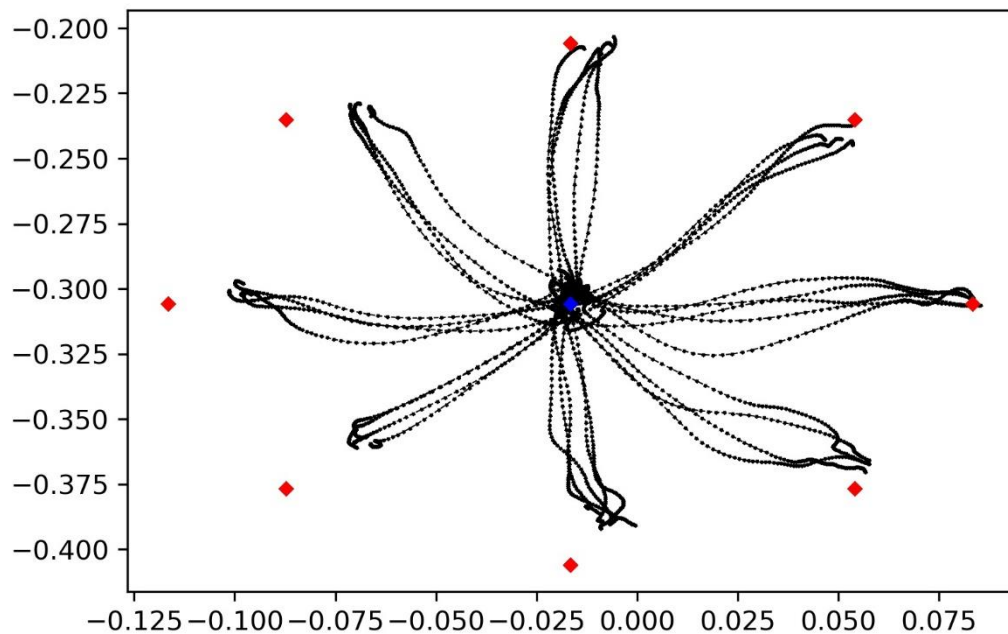
Figure 14 Center out task profile. The red dot indicated the center position of eight targets. The blue dot at the center indicated the home position. The black dashed line represents the reaching trajectory of the monkey C.

Figure 14 shows the behavioral data of the center-out task. The monkey didn't reach to the target completely because the target was 2 cm square. The total number of trials was 194. In the trials, success trials that acquired the target were analyzed. The number of trials that were analyzed is 175. We used the 60 % of total number of trials as the train data for the neural decoders. The remained 40 % was

equally divided into the validation and test data.

The silicon microelectrode array (96-channel, Blackrock, Inc) was implanted within the arm area of the contralateral primary motor cortex (M1). The total number of recorded neurons were 196. The neurons were sorted in descending order according to the number of firing rates and the neurons that made firing in the level of causing the singular matrix in the prediction process of the Kalman filter decoder were removed in the analysis. The 158 number of neurons are used.

2.2 Data analysis

2.2.1 Network design

In this section, the network structure of suggested neural decoder, which is speed-direction LSTM(SDLSTM), is explained. Also, velocity Kalman filter (VKF) and velocity LSTM (VLSTM) were compared with the SDLSTM to verify the performance of the new decoder. The reason of selecting those decoders as reference is that both Kalman filter and LSTM are state based decoder. The Kalman filter estimate the state with the observation of the neural data and the LSTM also estimate the hidden state including the state information from the neural data observation. The structure of VKF and VLSTM were copied from the works of Joshual I. Glaser (Joshual I. Glaser et al., 2017). The VKF had same algorithm with that implemented by W. Wu (W. Wu, 2003) except that it had scaling variable configuring the noise matrix of the kinematic states. We didn't configure the variable in our analysis because it was reported as the variable made only small improvements in the decoding quality of motor cortex signal. The firing rates in 50ms window consisted a bin. For each decoder, 3 bins were used as input.
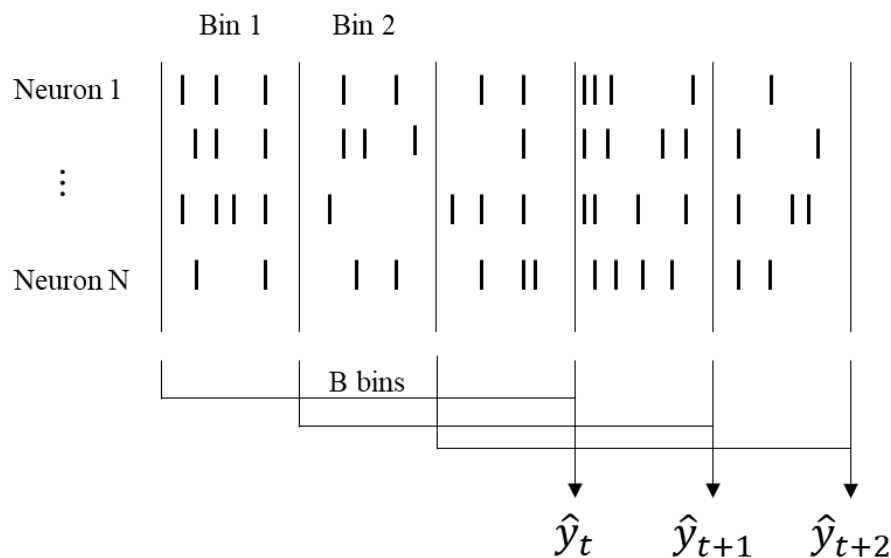
Figure 15 Input schematic for neural decoders

22

Figure 15 shows that each B number of bins were used to predict the kinematic variable at a specific time. Also, the window size was same with the size of time shift and thus, there were no overlap between each bin. Each bin is N dimensional vector representing the firing rates of N number of neurons (Figure 16).
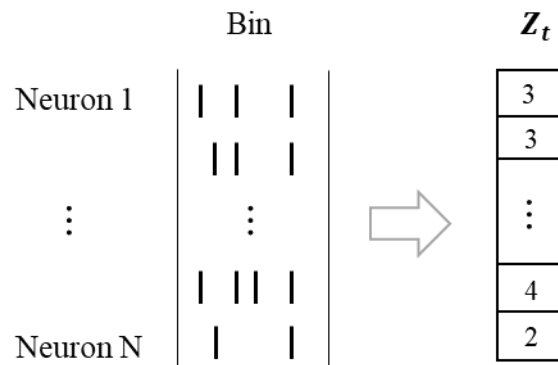
Figure 16 Firing rates in a bin of N number of neurons

2.2.1.1 Velocity Kalman Filter (VKF)
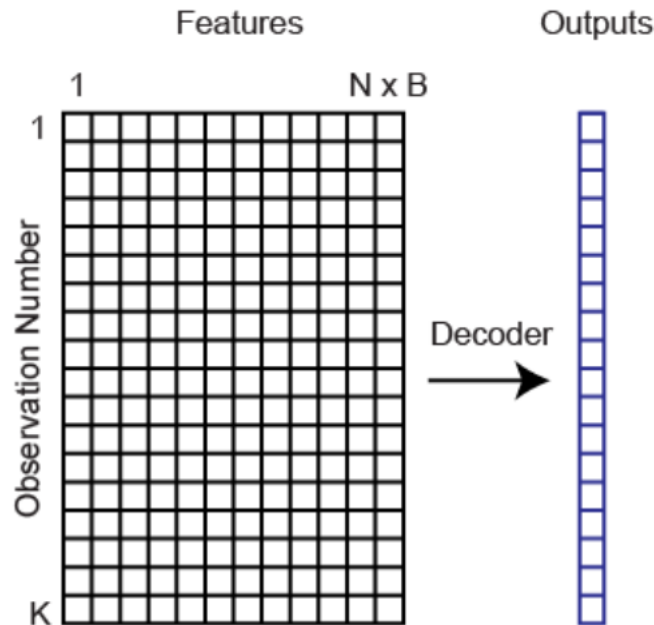
Figure 17 Input schematic for the velocity Kalman filter (Joshual I. Glaser et al., 2017)

The figure 17 show the input structure of the velocity Kalman filter. The N is the number of neurons and the B is the number of bins used for the decoder. Also, The K indicates the number of samples for train, validation or test. We used the 159 number of neurons and the 3 number of bins with 50ms window. Each bin had 159 dimension and it was concatenated as a single input for predict an observation. The 105 trials were used for train. The length of the train data was 2105. Also, the 35 trials were used for validation and test. The number of samples was 700 and 713 for validation and test. The number of features were 477, which was three times of the number of selected neurons. Also, although the velocity Kalman filter of the research of Joshual I. Glaser predicted position, velocity and acceleration, we only predicted the velocity parameter without prediction of the other parameter. The reason was that the other decoder, velocity LSTM and speed-direction LSTM, predicted only the velocity parameter. The figure 18 shows the structure of the velocity Kalman filter.
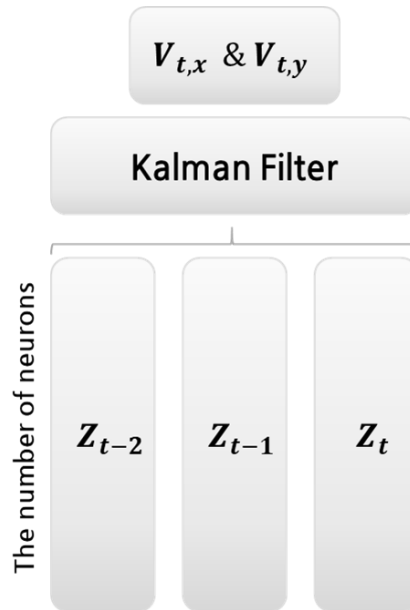
Figure 18 Structure of velocity Kalman filter

The Z_t is the number of firings of 158 neurons in 50ms time bin at the time t. The velocity Kalman filter got inputs concatenated from the time t-2 to the time t for predicting the $V_{t,x}$ and $V_{t,y}$.

2.2.1.2 Velocity LSTM (VLSTM)
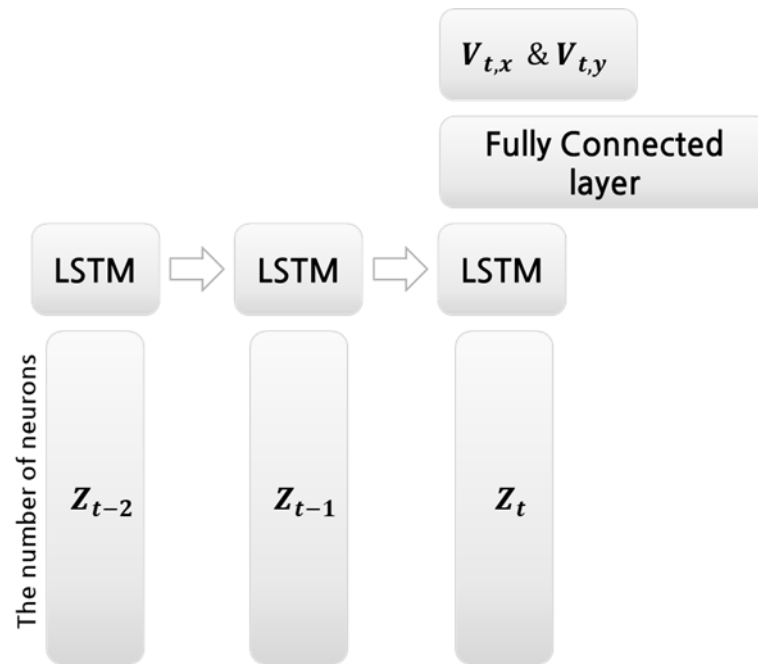
Figure 19 The structure of the velocity LSTM

Figure 19 represents the structure of the velocity LSTM. The Z_t is the number of firings of 158 neurons in 50ms time bin at the time t. The velocity LSTM got inputs serially from the time t-2 to the time t for predicting the $V_{t,x}$ and $V_{t,y}$. Each Z was processed in the LSTM cell and updated the previous cell state. It was passed to the LSTM cell of next time step. After processing the three number of Z sequentially, the final hidden state is passed to the fully connected layer which has two number of output units, the $V_{t,x}$ and $V_{t,y}$. We used the Adam optimizer to update the parameters in the network. The learning rate was 0.0001 and the β_1 was 0.9 and β_2 was 0.999. Also, the random search was used to find the optimal parameter set. The parameter of the number of hidden nodes, the batch size, the number of epoch was searched in specified boundary. The boundaries were from 10 to 50 for the number of hidden nodes and the batch size and from 10 to 100 for the number of epoch. The boundaries were selected heuristically.

2.2.1.3 Speed-Direction LSTM (SDLSTM)
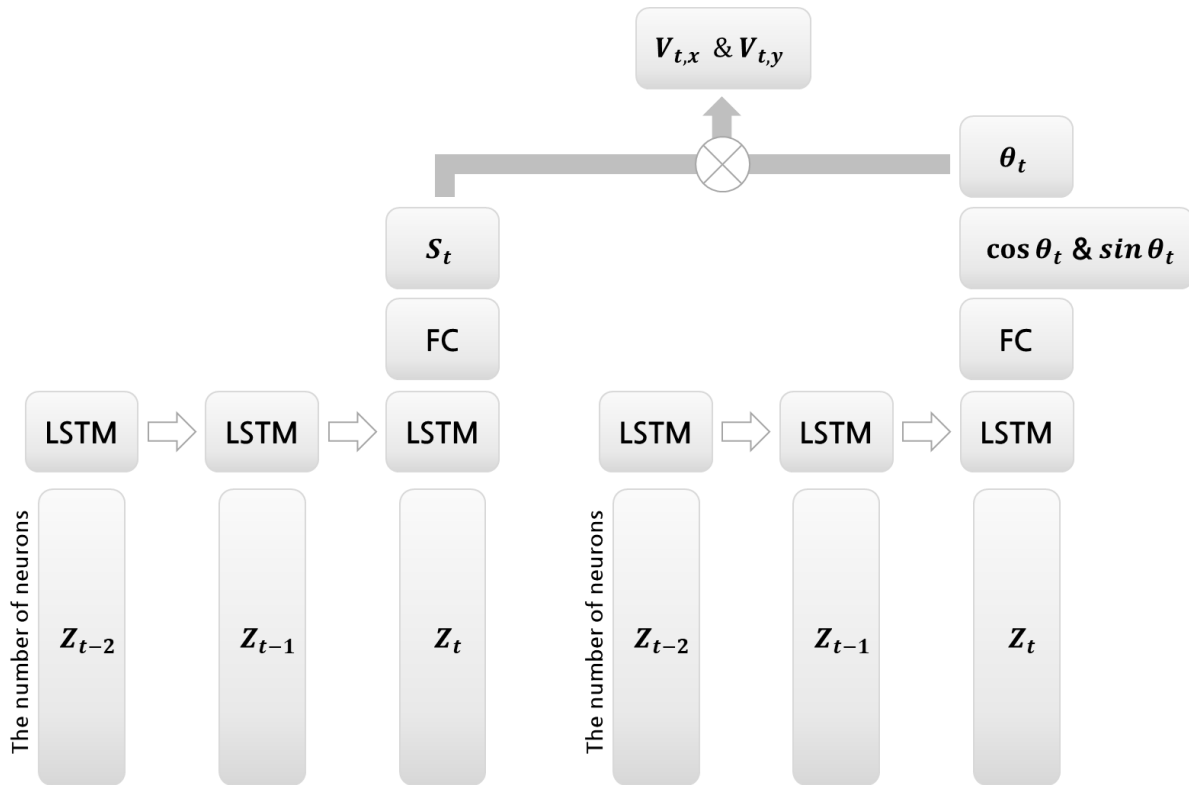
Figure 20 The structure of speed-direction LSTM

Figure 20 represents the structure of SDLSTM. The model has two parts of estimator. One part of the model estimates the arm movement speed and the other part estimates the arm movement direction by predicting the cosine and sine of the degree. Each model passed the hidden state to the fully connected layer which had one output unit for speed estimation and two number of output units for direction estimation. The velocity $V_{t,x}$ and $V_{t,y}$ were acquired with the estimated speed and degree. The reason that predicting cosine and sine instead of directly predicting the degree is because the degree has the property of periodicity. Because of the periodicity, the degree parameter has lots of discontinuous points in the recorded data while the trigonometrical function is continuous. These tricks have been used in previous researches (Rhys Heffernan, 2017 and James Lyons, 2014). When predicting the association angle of protein backbone, the deep neural-network had output units for predicting cosine and sine of the angle (James Lyons, 2014). Also, the bidirectional LSTM which had output units predicting cosine and sine for estimating the protein secondary structure was developed by Rhys Heffernan (Rhys Heffernan, 2017). The learning rate was 0.0001 and the β_1 was 0.9 and β_2 was 0.999. Also, the random search was used to find the optimal parameter set. The parameter of the number of hidden nodes, the batch size, the number of epoch was searched in specified boundary. The boundaries were from 10 to 50 for the number of hidden nodes and the batch size and from 10 to 100 for the number of epoch. The boundaries were selected heuristically.

2.2.2 Measurement index

The quality of the reconstruction need to be measured quantitively to measure how accurate the neural decoders were. The measurement index of the reconstructed velocity and position is introduced in this section. All the indexes were averaged The measurement indexes of angular difference, correlation coefficient, mean absolute error, Euclidean distance were averaged on each trial and then those averaged indexes are compared according to the decoding method. The repeated measure ANOVA (rmANOVA) was performed on the indexes of each decoders. Also, the Mauchly's test was used to check the violation of the sphericity. If the sphericity was violated, the significance of the rmANOVA was corrected by Greenhouse-Geisser method if the epsilon estimated by the Greenhouse-Geisser method was less than 0.75, the p-value of rmANOVA corrected by Greenhouse-Geisser method was used and if the epsilon was larger than 0.75, the p-value of rmANOVA corrected by Huynd-Feldt method was used. When the p-value was corrected, the epsilon was noticed as ε. Also, the multiple comparison test corrected by bonferonni was performed as a *post hoc* test.

2.2.2.1 Measuring the accuracy of velocity reconstruction

2.2.2.1.1 Angular difference

Angular difference (AD) is the difference of the angle between true velocity vector and predicted velocity vector in radian. It measures the quality of the directional information decoding.
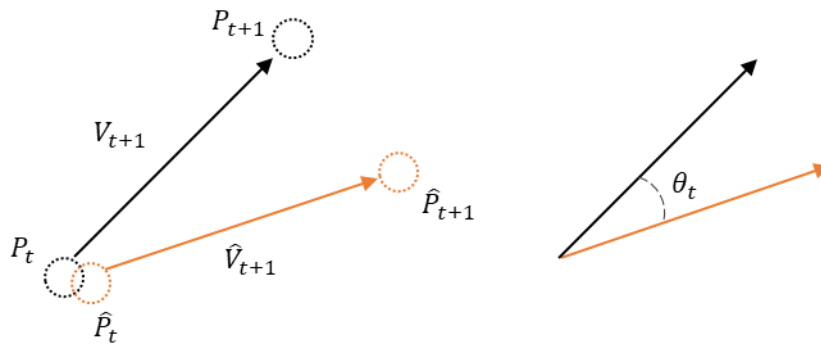
Figure 21 Angular difference between predicted and behavioral data

In the diagram, the V_{t+1} is true velocity vector at time t+1, P_{t+1} is true position at time t+1, $\hat{V}_{t+1}$ is the decoded velocity vector at time t+1, $\hat{P}_{t+1}$ is the decoded position at time t+1. Using the following equation, θ_{t+1} can be acquired.

$$\theta_{t+1} = \cos^{-1}\frac{V_{t+1} \cdot \hat{V}_{t+1}}{|V_{t+1}| \, |\hat{V}_{t+1}|} \tag{20}$$

The averaged AD of a single trial is acquired with following equation:

$$averaged\ AD = \frac{1}{n}\sum_{t=0}^{n}\theta_t \tag{21}$$

The n is the number of samples in a trial.

2.2.2.1.2 Correlation coefficient

Pearson correlation coefficient(CC) is a measure of linear correlation of two variables, X and Y. It measures the quality of the velocity reconstruction. It is acquired with the covariance of the two variables.

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X\sigma_Y} \tag{22}$$

The $cov(X,Y)$ is the covariance between X and Y. The σ_X and σ_Y is the standard deviation of X and Y. It was acquired in a single trial with following equation:

$$r = \frac{\sum_{i=1}^{n}(v_{x,i} - \overline{v_x})(v_{y,i} - \overline{v_y})}{\sqrt{\sum_{i=1}^{n}(v_{x,i} - \overline{v_x})^2}\sqrt{\sum_{i=1}^{n}(v_{y,i} - \overline{v_y})^2}} \tag{23}$$

The n is the number of samples in a trial.

2.2.2.1.3 Mean absolute error

The mean absolute error (MAE) indicates the averaged error of the velocity prediction. It was acquired in a single trial with following equation:

$$MAE\ of\ velocity\ x\ and\ y = \frac{MAE\ of\ velocity\ x + MAE\ of\ velocity\ y}{2} \tag{24}$$

$$MAE\ of\ velocity\ x = \frac{\sum_{i=1}^{n}|v_{x,i} - \hat{v}_{x,i}|}{n} \tag{25}$$

$$MAE\ of\ velocity\ y = \frac{\sum_{i=1}^{n}|v_{y,i} - \hat{v}_{y,i}|}{n} \tag{26}$$

The n is the number of samples in a trial.

2.2.2.2 Measuring the accuracy of position reconstruction

2.2.2.2.1 Euclidian distance with true trajectory

$$\text{ED} = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(p_{x,i} - \hat{p}_{x,i})^2 + (p_{y,i} - \hat{p}_{y,i})^2} \qquad (27)$$

The Euclidean distance with true trajectory measures that how accurately the trajectory was reconstructed by each decoder. . It was acquired in a single trial. The n is the number of samples in a trial. $p_{x,i}$ is the position of x axis at i-th sample. $\hat{p}_{x,i}$ is the reconstructed position of x axis at the sample.

2.2.2.3 Measuring the effectiveness of position reconstruction

Although the reconstructed trajectory is not completely accurate, the decoded trajectory will be meaningful if it is efficient to acquire a given target. The following indexes measure that kind of aspects of the decoding results.

In the BMI paradigm, interpreting the neural information accurately is important because it's final goal is making the interface that transfer the subject's intention to the device that wants to control without distortion and making natural movement with the interface. However, if the interface could not fully interpret the intention, it can be another important aspect for evaluating the BMI interface that how effectively the BMI helps to achieve the goal of the intention such as reaching to the target direction.

2.2.2.3.1 Euclidian distance with ideal trajectory

The Euclidean distance (ED) with ideal trajectory represents how efficiently the reconstructed movements reached to the target. When the hand moves to the target, it can be said that the target was acquired in most efficient movement if the trajectory is on the straight line connecting between the home and target. The straight line is the ideal trajectory. The Figure 22 represents the concept of the index. The orange line represents the reconstructed trajectory and the black line indicates the ideal trajectory. The dashed black line shows the distance between the position of the trajectory and the straight line.
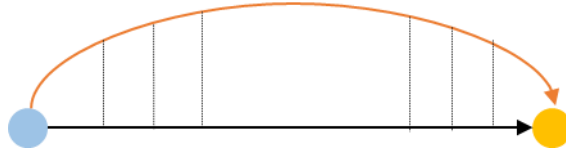
Figure 22 Euclidean distance with ideal trajectory

The index was acquired on each trial and compared for each decoding method. For each trial, the following equation was used.

$$ED \text{ with ideal trajectory} = \frac{1}{n}\sum_{i=1}^{n}\sin(\cos^{-1}(\frac{\hat{p}_i \cdot d}{|\hat{p}_i||d|})) \qquad (28)$$

$$\hat{p}_i = (\hat{p}_{x,i} - p_{x,home}, \hat{p}_{y,i} - p_{y,home}) \qquad (29)$$

$$d = (p_{x,target} - p_{x,home}, p_{y,target} - p_{y,home}) \qquad (30)$$

The $\hat{p}_{x,i}$, $\hat{p}_{y,i}$ is the predicted x and y position of i-th sample. The $p_{x,target}$, $p_{y,target}$ is the x and y position of given target. The $p_{x,home}$, $p_{y,home}$ is the x and y position of home. With the angle between the $\hat{p}_i$ and d, the length of the perpendicular line from the decoded position to the straight line was calculated.

2.2.2.3.2 Hit rate

The hit rate index measure how many times the movement trajectory reached to the target. The square box surrounding the target that has edge length of 0.4 cm was accepted as safe boundary for a hit trail. The hit rate was acquired by dividing the number of hit trials with the total number of trials.

2.2.2.3.3 Distance to target according to time

This index measure how accurately the reconstructed trajectory approach to the target in a trial. If the trajectory is predicted accurately, the graph of distance according the time is similar with that of the true trajectory. Because the time consumed by a trial was different between trials, the distance was measured

for every 10 % of the duration consumed in each trial. The collected distances on each percentage were averaged and the standard error was indicated.

2.2.2.3.4 Movement directional change (MDC)

The MDC index measure the smoothness of the reconstructed trajectory. It counts how many the trajectory changes the movement direction relative to the straight line between home and target position. The MDC value is larger than 0. If a trajectory has small MDC, it means that the trajectory has smooth path. It was acquired in a single trial.

Figure 23 Movement Direction Change

2.2.2.3.5 Orthogonal directional change (ODC)

The ODC index also measure the smoothness of the reconstructed trajectory. However, it counts the direction change parallel to the orthogonal line of the straight line between the home and target. The ODC value is larger than 0. If a trajectory has small ODC, it means that the trajectory has smooth path. The MDC and ODC index were calculated with at least 100ms and 200ms interval because we don't want to regard the oscillating trajectory in short time step as the directional change. It was acquired in a single trial.
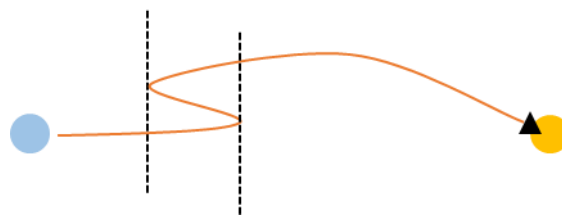
Figure 24 Orthogonal direction change

3. Results

THIS PAGE INTENTIONALLY LEFT BLANK

3.1 Reconstructed velocity

Velocity Kalman Filter (VKF)

Speed Direction LSTM (SDLSTM)

Velocity LSTM (VLSTM)
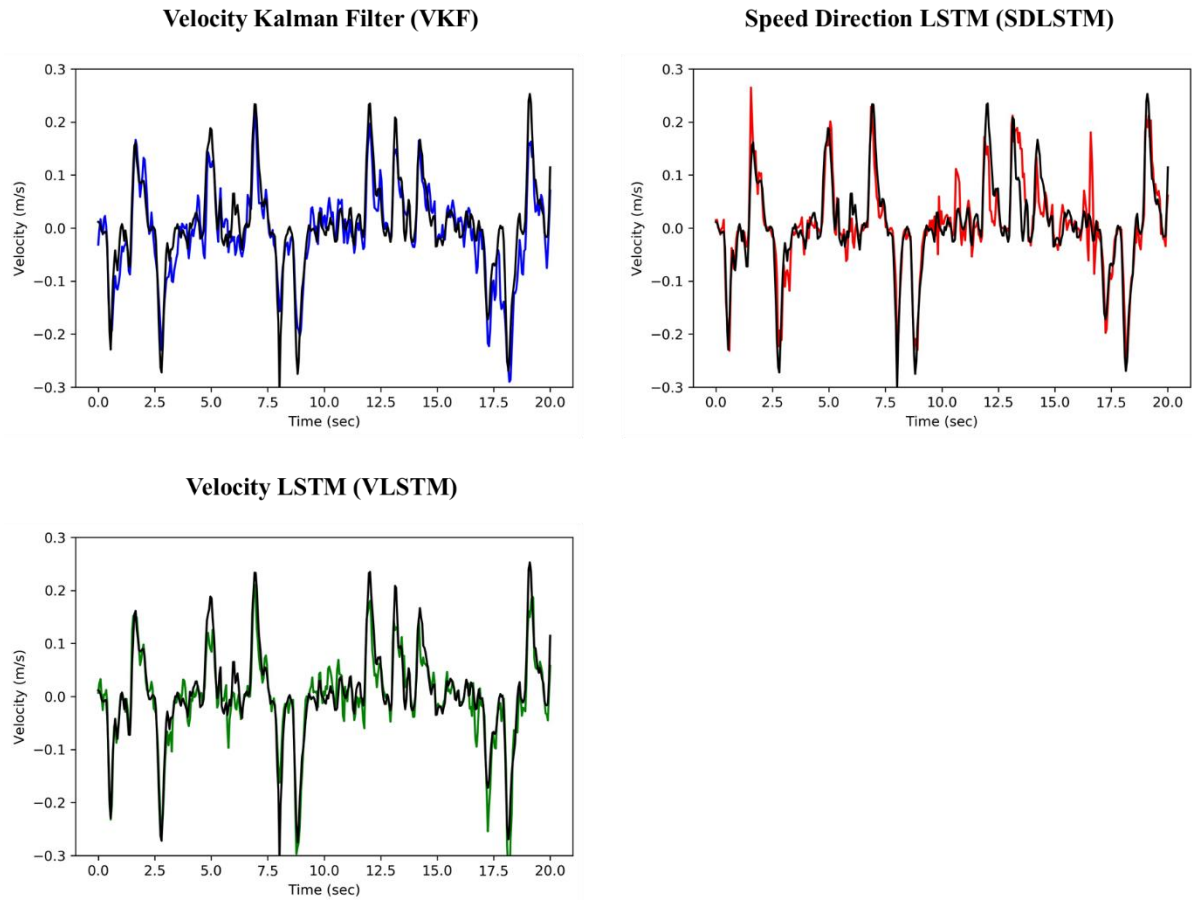
Figure 25 An example of velocity x profile in the test data reconstructed from each neural decoder. The true velocity profile was indicated as a black line.

The figure 25 shows that an example of the velocity x profile in test data predicted from each neural decoder. All decoders look like following the true velocity profile. The SDLSTM showed better prediction in the peak of velocity while the other decoders made insufficient prediction to follow the peak velocity of the true data. However, the SDLSTM showed poor prediction than the other decoders when the amplitude should be small.

Velocity Kalman Filter (VKF)

Speed Direction LSTM (SDLSTM)

Velocity LSTM (VLSTM)
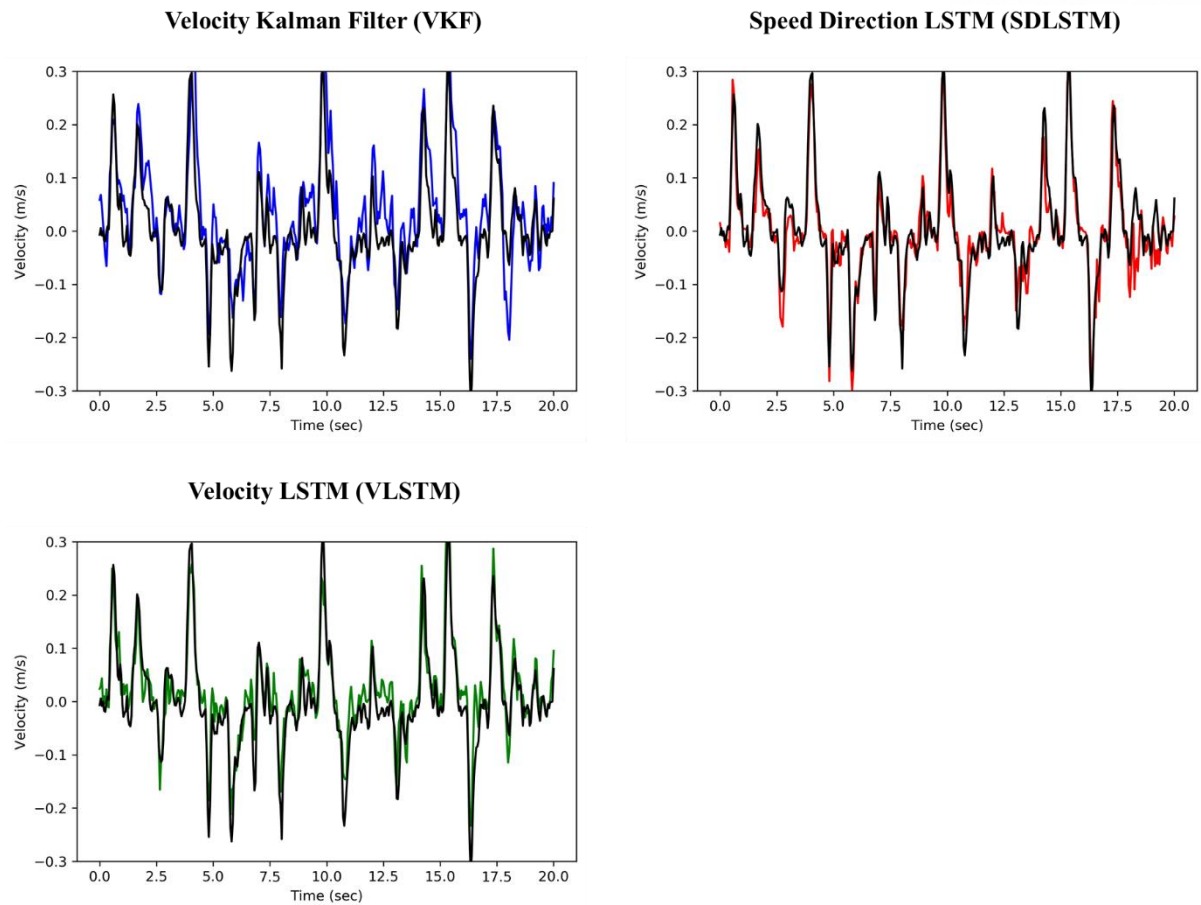
Figure 26 An example of velocity y profile in the test data reconstructed from each neural decoder. The true velocity profile was indicated as a black line.

The figure 26 shows the example profile of the velocity y reconstructed by each neural decoder. The SDLSTM and VLSTM predicted the bell shape of velocity in high accuracy while the KF showed noisy prediction at the same moment. Also, the SDLSTM predicted the velocity with less error when the velocity should be small while it was noisy in the prediction of x velocity.

3.2 Reconstructed trajectory
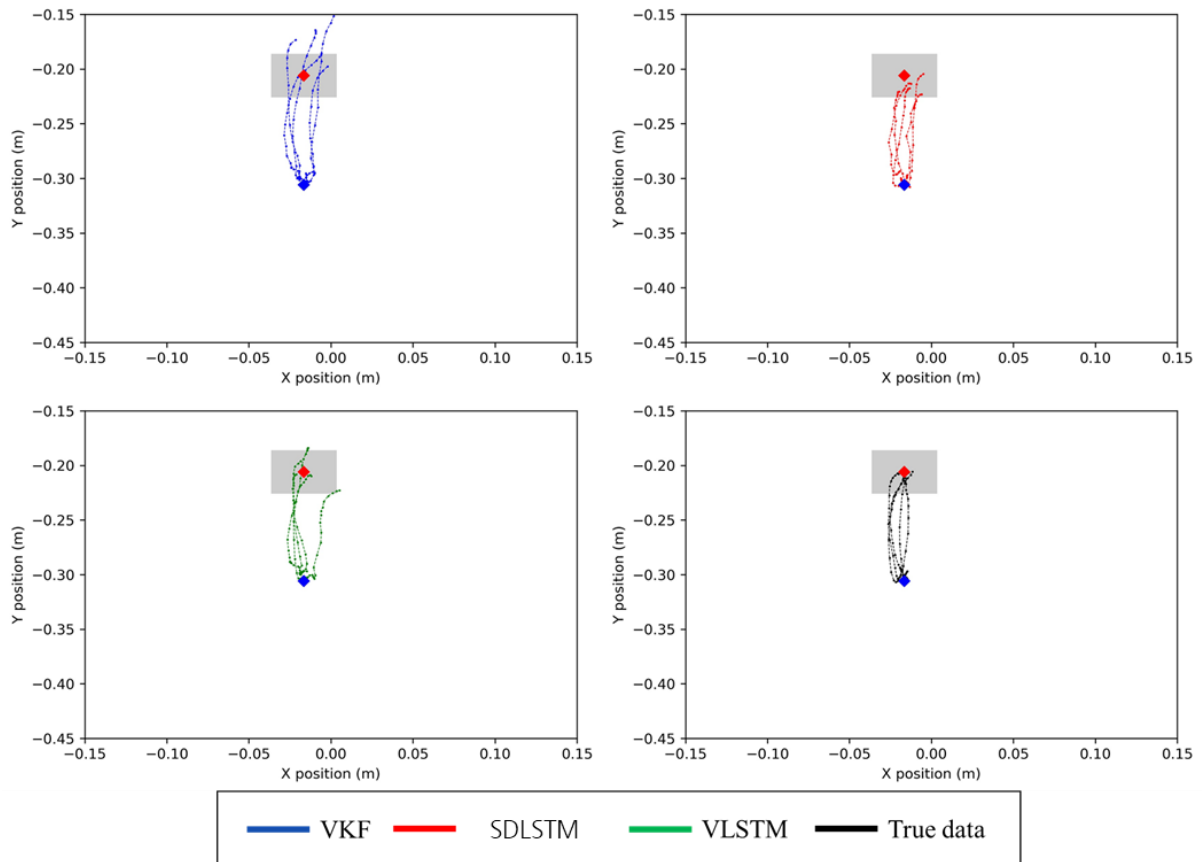
Figure 27 An example of reconstructed trajectory and the true trajectory in the test data. The target was indicated as red diamond and the home was indicated as the blue diamond. The shaded box implied the safe area that if the trajectory ends in the boundary, it was regarded as the success trial.

Figure 27 shows an example of the reconstructed trajectory of trials to the target direction of -45 degrees. The trajectory reconstructed by the VKF and the VLSTM didn't reach to around the target. In the other hands, that of the SDLSTM reached to the end of the shaded box although it didn't the acquire the target.

3.3 Measuring the accuracy of velocity reconstruction

3.3.1 Angular difference
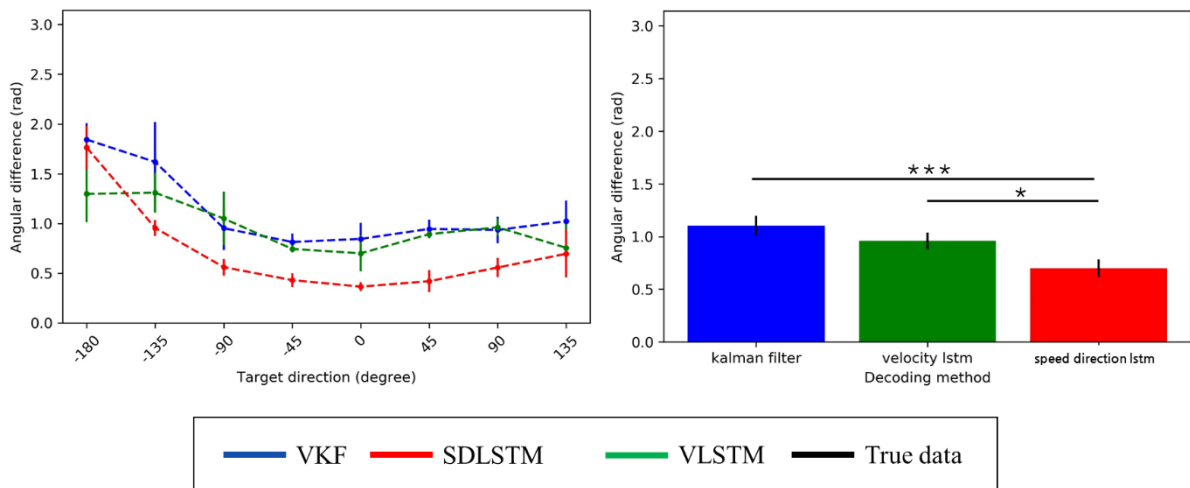
Figure 28 Angular difference in the movement direction. (Left) the averaged AD according to the target direction. (Right) the averaged AD of the whole test data. Standard errors were indicated as a thin bar.

The figure 28 shows that the angular difference (AD) of each decoder. The SDLSTM predicted the movement direction in higher accuracy that the other decoders at most of the target direction (figure 28, left). Specifically, the one-way repeated measure ANOVA on the averaged AD of the whole test data revealed that the effect of decoders were significant, $F(2,68) = 12.683$, $p < 0.001$. The *post hoc* test corrected with bonferroni revealed that angular difference of the VKF (M= 1.104, SE = 0.0943) was not significantly different that of the VLSTM (M = 0.9587, SE = 0.0808) at $p = 0.13544$. The AD of SDLSTM (M = 0.6994, SE = 0.0852) was significantly different with that of the VKF at $p<0.001$ and with that of VLSTM at $p<0.05$

3.3.2 Correlation coefficient

Figure 29 Correlation coefficient of the reconstructed velocity on the x axis. (Left) the averaged CC according to the target direction. (Right) the averaged CC of the whole test data. Standard errors were indicated as a thin bar.

It can be found that the CC of the velocity on the x axis was low in several specific direction (Figure 29). For example, CC of all decoders became poor at the target direction of -90 degree and 90 degrees. The CC in the other direction were similar between the decoders (Figure 29, left). Thus, the CC averaged on the whole trial became affected by the direction of -90 and 90 degree and the VLSTM showed best CC in the velocity prediction in the x axis. Specifically, the one-way repeated measure ANOVA on the averaged CC of the whole test data revealed that the effect of decoders was marginally significant, $F(2,68) = 2.6833$, $p = 0.076$. Further, the *post hoc* tests corrected with bonferroni revealed that the CC of VKF ($M = 0.8047$, $SE = 0.0408$) was not significantly different with that of VLSTM ($M = 0.813$, $SE = 0.0353$) at $p = 1$. Also, that of VKF and that of SDLSTM ($M = 0.755$, $SE = 0.0464$) was not significantly different with each other at $p = 0.33267$. The CC of VLSTM was marginally significant different with that of SDLSTM at $p = 0.1033$.

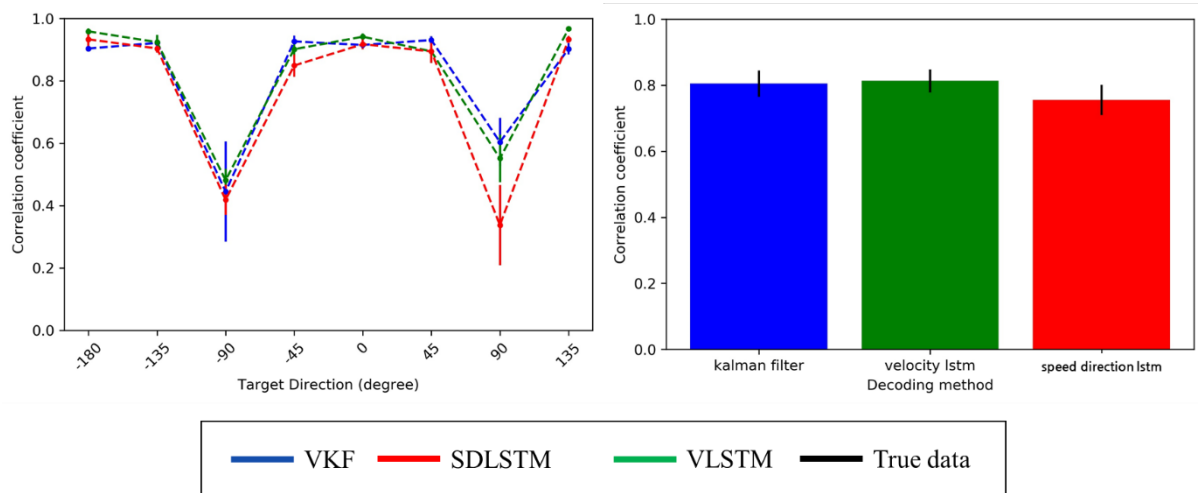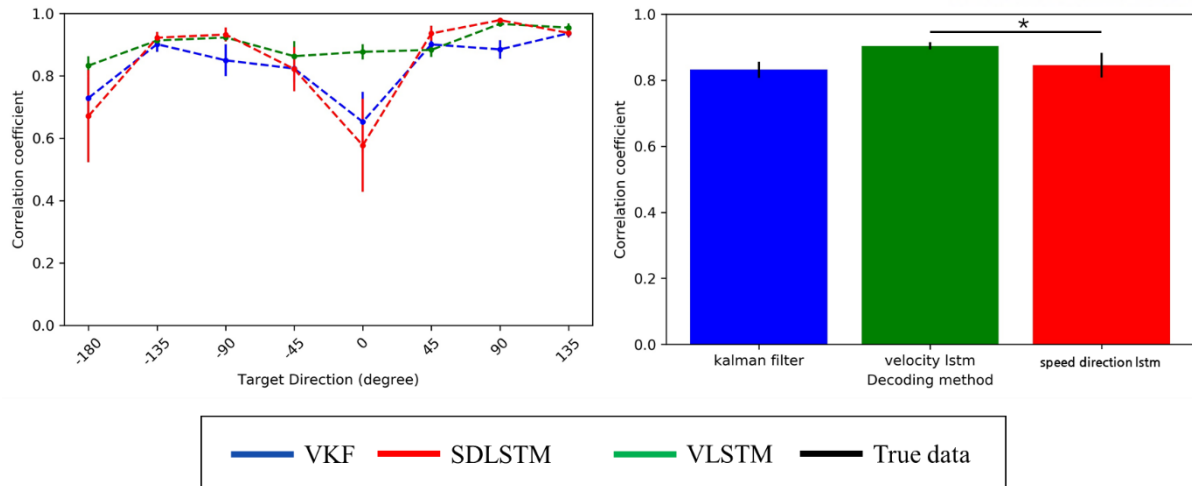Figure 30 Correlation coefficient of the reconstructed velocity on the y axis. (Left) the averaged CC according to the target direction. (Right) the averaged CC of the whole test data. Standard errors were indicated as a thin bar.

The CC of y velocity was poor in the SDLSTM and the VKF at the target direction of 0 degree. Specifically, the one-way repeated measure ANOVA on the averaged CC of the whole test data revealed that the effect of decoders was marginally significant, $F(2,68) = 3.057$, $p = 0.059$, $\varepsilon = 0.898$. The HuynhFeldt correction was used. Further, the *post hoc* tests corrected with bonferroni revealed that there was significant difference between that of VKF ($M = 0.8316$, $SE = 0.0249$) and that of VLSTM ($M = 0.9036$, $SE = 0.0111$) at $p < 0.05$. The difference between the VLSTM and VKF was not significant at $p = 1$. Also, the difference between the VLSTM and the SDLSTM ($M = 0.8458$, $SE = 0.038$) was not significant at $p = 0.30815$.

3.3.3 Mean absolute error

Figure 31 Mean absolute error of predicted velocity on x axis. (Left) the averaged MAE according to the target direction. (Right) the averaged MAE of the whole test data. Standard errors were indicated as a thin bar.

The mean absolute error on x axis was smallest at the VLSTM. Specifically, the one-way repeated measure ANOVA on the averaged MAE of the whole test data revealed that the effect of decoders were significant, $F(2,68) = 4.9134$, $p < .05$, $\varepsilon = 0.795$. The HuynhFeldt correction was used. Further, the *post hoc* tests corrected with bonferroni revealed that there was significant difference between that of VKF ($M = 0.028$, $SE = 0.0019$) and that of VLSTM ($M = 0.0231$, $SE = 0.0012$) at $p < .01$. The SDLSTM ($M = 0.755$, $SE = 0.0464$) didn't showed any significant difference with the other decoders.
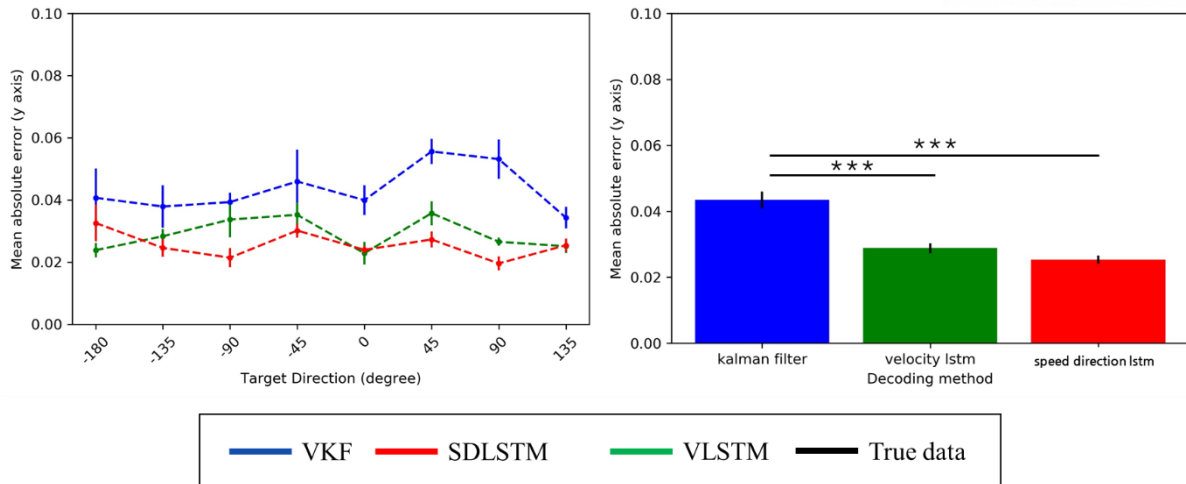
Figure 32 Mean absolute error of predicted velocity on y axis. (Left) the averaged MAE according to the target direction. (Right) the averaged MAE of the whole test data. Standard errors were indicated as a thin bar.

The mean absolute error on y axis was smallest at the SDLSTM. Specifically, the one-way repeated measure ANOVA on the averaged MAE of the whole test data revealed that the effect of decoders were significant, $F(2,68) = 32.536$, $p < .001$, $\varepsilon = 0.87$. The HuynhFeldt correction was used. Further, the post hoc tests corrected with bonferroni revealed that there was significant between the VKF (M = 0.0434, SE = 0.0026) and the VLSTM (M = 0.0288, SE = 0.0015) at $p < .001$. The VKF showed significant difference with the SDLSTM (M = 0.0253, SE = 0.0013) at $p < .001$.
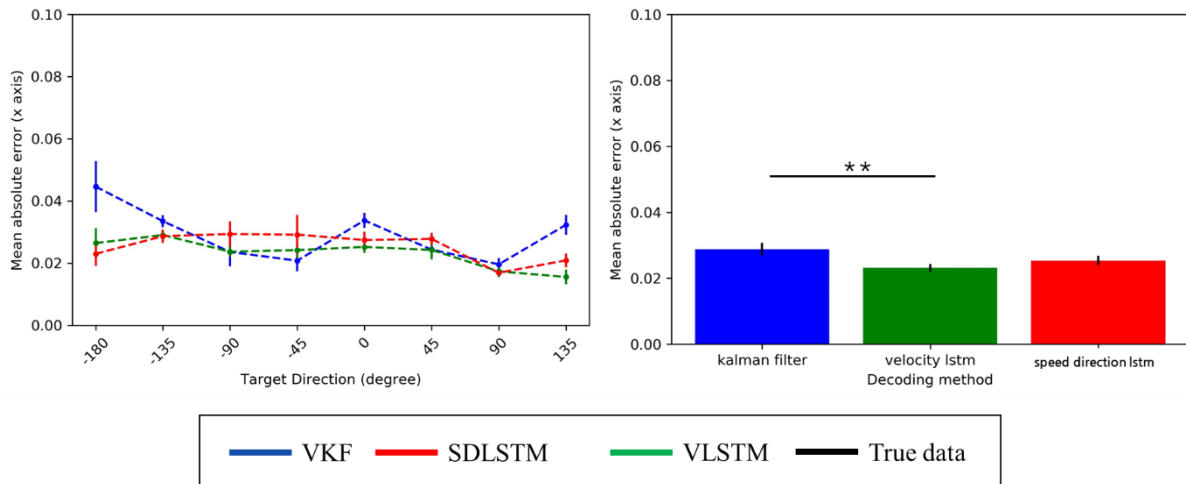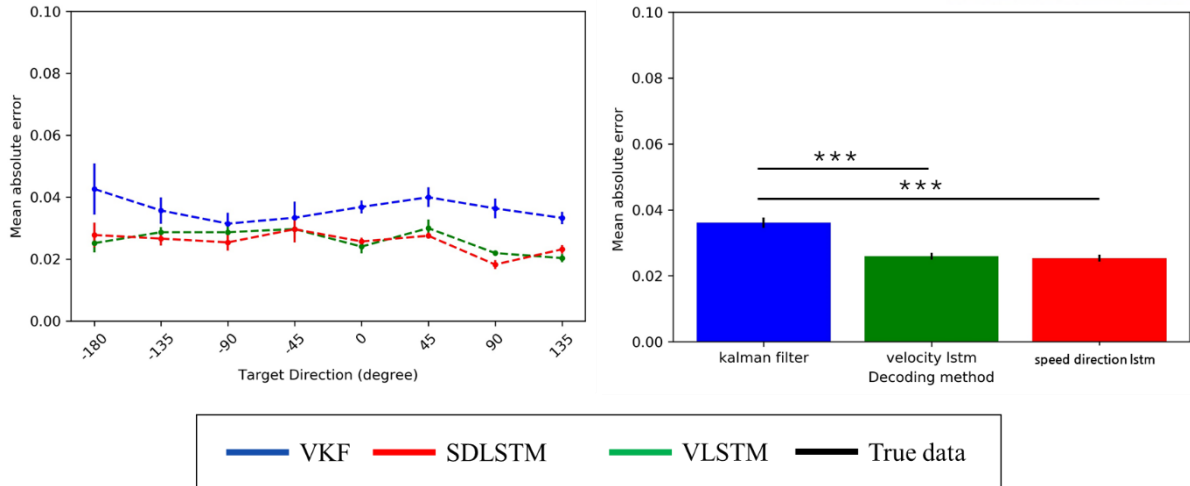
Figure 33 Mean absolute error of predicted velocity averaged on x and y axis. (Left) the averaged MAE according to the target direction. (Right) the averaged MAE of the whole test data. Standard errors were indicated as a thin bar.

To figure out the overall correctness of the velocity prediction, the averaged MAE on the both axis was compared between the decoders. The results showed that the SDLSTM and VLSTM were in similar level of the index and they were better than the that of VKF. Specifically, the one-way repeated measure ANOVA on the averaged MAE of the whole test data revealed that the effect of decoders were significant, $F(2,68) = 34.73$, $p < .001$, $\varepsilon = 0.758$. The HuynhFeldt correction was used. Further, the post hoc tests corrected with bonferroni revealed that there was significant difference between the VKF (M = 0.0361, SE = 0.0016) and VLSTM (M = 0.026, SE = 0.001) at $p < .0001$ and between the VKF and the SDLSTM (M = 0.0253, SE = 0.0011) at $p < .0001$ in the MAE. The difference between the VLSTM and the SDLSTM was not significant.

3.4 Measuring the accuracy of position reconstruction

3.4.1 Euclidean distance with true trajectory
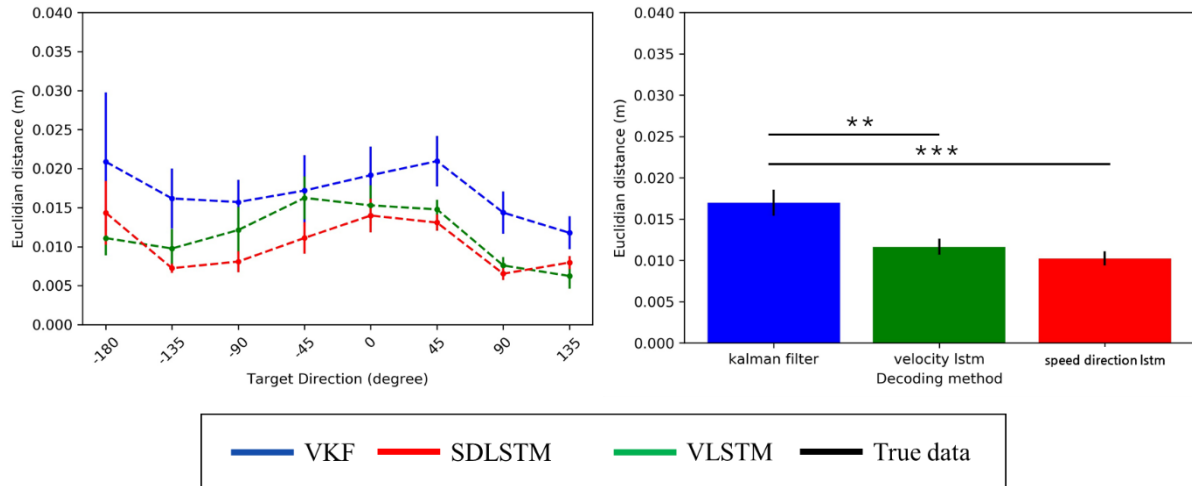
Figure 34 Averaged Euclidean distance between true and reconstructed trajectory. Standard errors were indicated as a thin bar. Red (Left) the averaged ED according to the target direction. (Right) the averaged ED of the whole test data.

Figure 34 shows the averaged Euclidean distance of each decoders. Specifically, the one-way repeated measure ANOVA on the averaged ED of the whole test data revealed that the effect of decoders were significant, $F(2,68) = 14.726$, $p < 0.0001$. The *post hoc* test corrected with bonferroni revealed that the Euclidean distance of VKF (M = 0.017, SE = 0.0016) was significantly higher than that of VLSTM (*M* = 0.0118, SE = 0.0011) at $p < .01$ and that of the VKF was significantly higher than that of SDLSTM (M= 0.0103, SE = 8.4455e-04) at $p < .0001$. However, the difference between the VKF and SDLSTM was not significant at $p = .48712$.

3.5 Measuring the effectiveness of position reconstruction

3.5.1 Euclidean distance with ideal trajectory
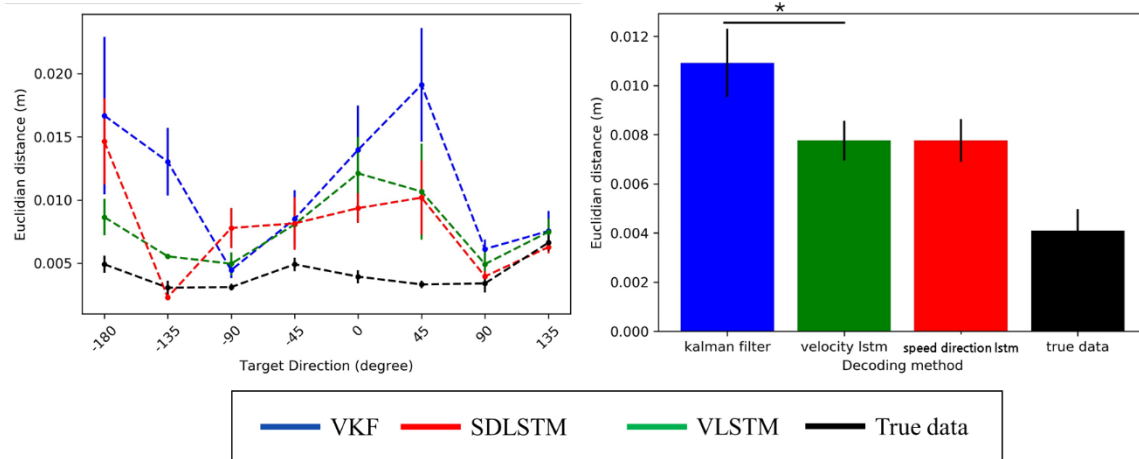
Figure 35 Averaged Euclidean distance with ideal trajectory of the reconstructed trajectory. (Left) the averaged ED according to the target direction. (Right) the averaged ED of the whole test data. Standard errors were indicated as a thin bar.

The reaching movement was close to the ideal straight line to the target although the trajectory had curved shape which was natural movement of reaching. In the other hand, the decoded trajectories were far from the ideal. Specifically, the distances were bigger in some direction such as -180 degrees or 45 degrees and were became small in the neighboring direction of the direction. this pattern could be observed in the Euclidean distance with true trajectory. It implies that the error in reconstructing the trajectory made the trajectory far from the ideal straight line. Specifically, the one-way repeated measure ANOVA on the averaged ED of the whole test data revealed that the effect of decoders was not significant, $F(2,68) = 4.7436$, $p < 0.05$. Further, the *post hoc* tests corrected with bonferroni revealed that the ED of VLSTM (M = 0.0078, SE = 8.2349e-04) was significantly lower than that of VKF (M = 0.0109, SE = 0.0014) at $p < 0.05$. The difference between that of the VKF and the SDLSTM (M = 0.0078, SE = 8.8148e-04) was not significant (p = 0.087). Also, the difference between the VLSTM and the SDLSMT was not significant (p = 1). Also, the true data has mean of 0.004 and standard error of 2.777e-04.

3.5.2 Hit rate
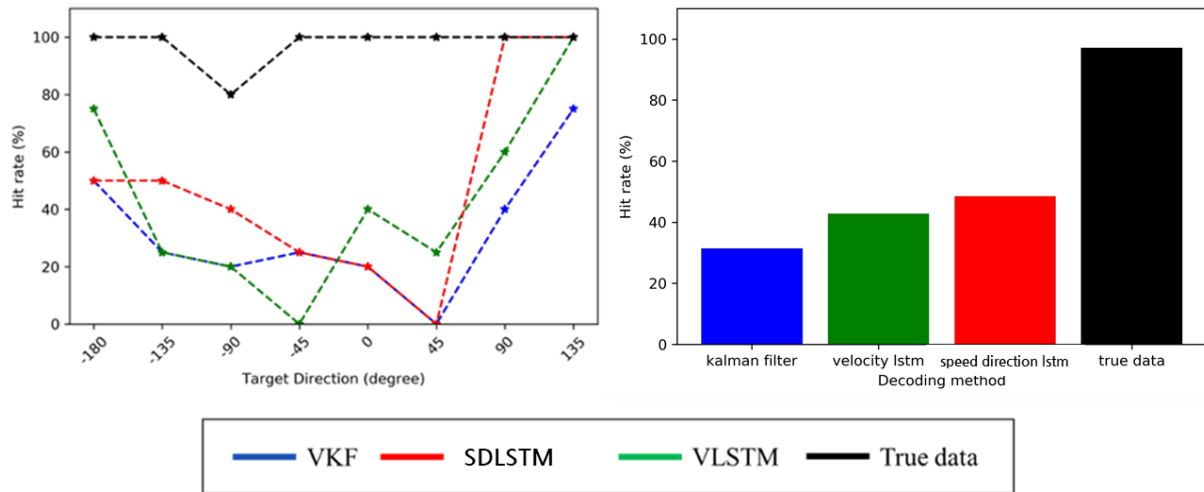
Figure 36 Hit rate of the reconstructed trajectory. (Left) the hit rate according to the target direction. (Right) the hit rate of the whole test data.

The hit rates performance was in the order of SDLSTM, VLSTM and VKF. At most of the target directions, the SDLSTM was better than the other decoders. The hit rates in 90 and 135 degrees were 100% in the SDLSMT.
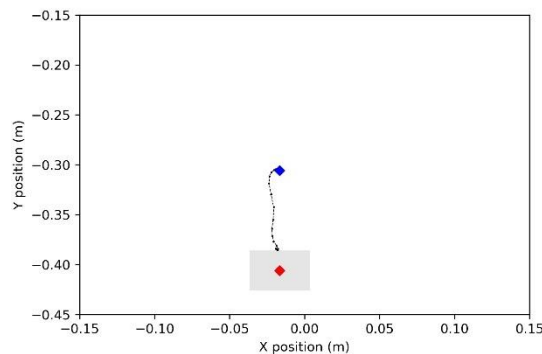
Figure 37 The trial fail to reach the target, but marked as true trial

The Figure 37 showed the reason that the hit rate of the behavioral data was not 100 %. The monkey didn't fully reach to the target because he knew that he only had to put the endpoint of hist arm in the boundary box around the target in which the trial is regarded as a hit trail. It made the most of reaching finish near the edge of the boundary box. An example of trajectory ends before it safely goes into the box although it was marked as hit trial (see Figure 37).

3.5.3 Distance to target according to time
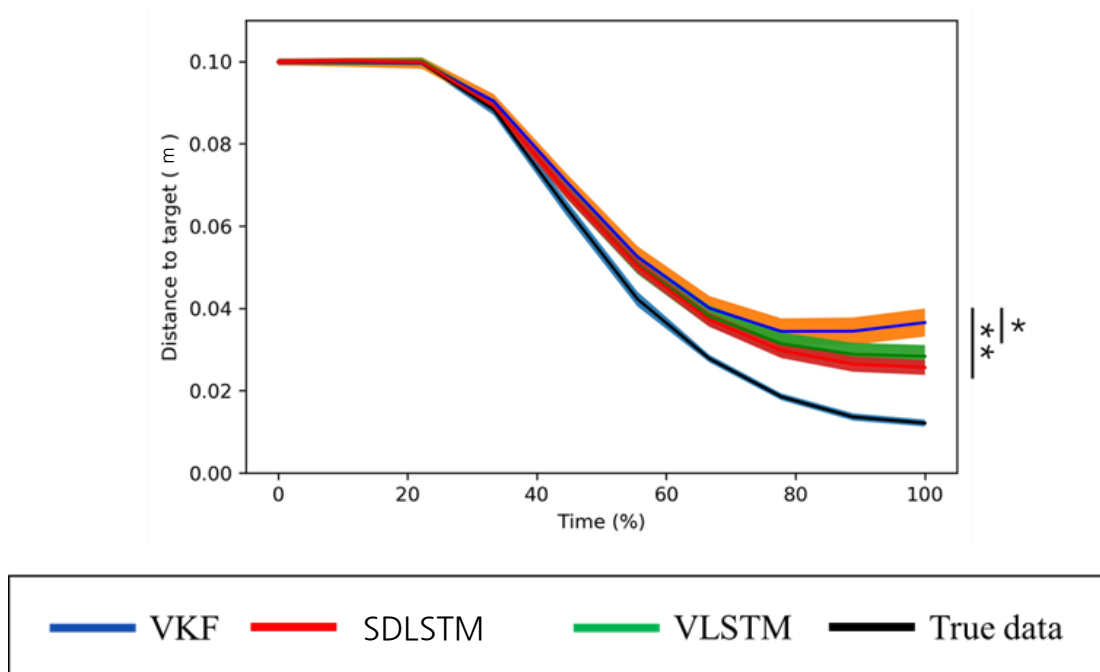
Figure 38 Distance to target according to task time percentage. The standard error of each decoder was indicated as a filled color which is same with that of each decoder.

The figure 38 shows that the SDLSTM and the VLSTM approached to the target faster than the VKF implying the decoders' accuracy of trajectory reconstruction was better than that of the VKF. The reason that the distance of the VKF became larger after 80 % progress of the trial is that the decoded trajectory of the VKF frequently go through the target. The statistical test was performed only at the 100 % of the time progress. Specifically, the one-way repeated measure ANOVA on the distance to the target of the whole test data revealed the effect of decoders on the index, $F(2,68) = 8.556$, $p < .001$. The *post hoc* tests corrected with bonferroni revealed that there was no difference between that of SDLSTM (M = 0.0256, SE = 0.0018) and that of VLSTM (M = 0.0284, SE = 0.0028) at p = 1. The distance of VKF (M = 0.0366, SE = 0.0034) was significantly different with that of the VLSTM at p< .05 and that of SDLSTM at p< .01. Also, the true data has mean of 0.012 and standard error of 8.56e-04.

3.5.4 Movement directional change (MDC)
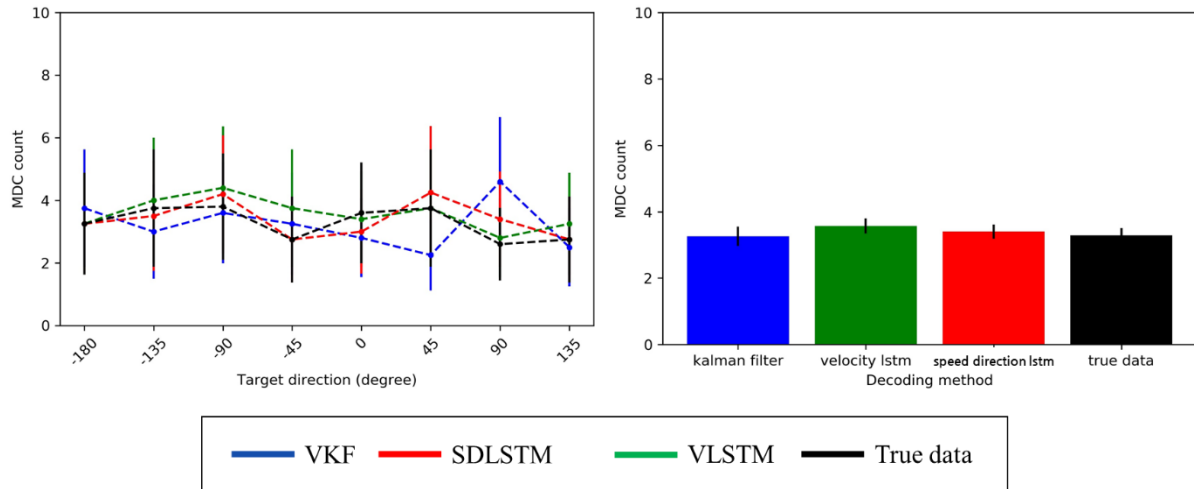
Figure 39 Movement direction change of reconstructed trajectory. * indicates the significant difference between two decoders.

The figure 39 showed the MDC in each neural decoder. Specifically, the one-way repeated measure ANOVA on the MDC of the whole test data didn't revealed the effect of decoders, $F(2,68) = 0.516$, $p = .60$. The *post hoc* tests corrected with bonferroni revealed that there was no difference between the neural decoders. The significance level was $p = 1$ in the all combination among the VKF (M = 3.257, SE = 0.294), VLSTM (M = 3.571, SE = 0.233) and SDLSTM (M = 3.4, SE = 0.214). The behavioral data showed mean of 3.857 and standard error of 0.227.

3.5.5 Orthogonal directional change (ODC)
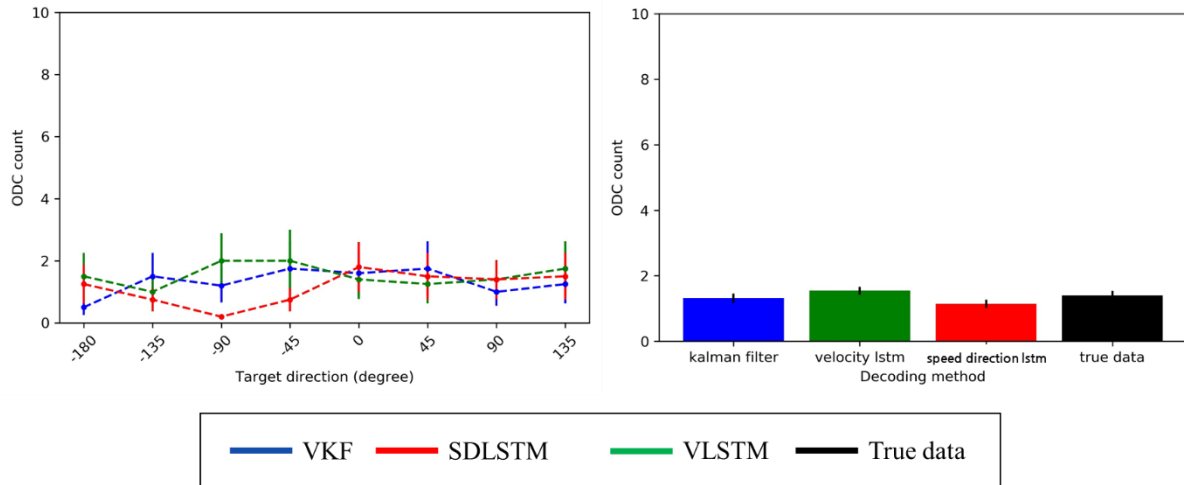
Figure 40 Orthogonal direction change of reconstructed trajectory. * indicates the significant difference between two decoders.

The figure 40 showed the ODC in each neural decoder and the SDLSTM showed most smooth trajectories in the orthogonal direction to the target. Also, VKF was better than the VLSTM in the index. Specifically, the one-way repeated measure ANOVA on the ODC of the whole test data revealed the marginally significant effect of decoders, $F(2,68) = 2.777$, $p = .069$. The *post hoc* tests corrected with bonferroni revealed that there was no difference between the VKF (M = 1.3143, SE = 0.1465) and the other decoders, in which the p-value was 0.44 at the comparison with that of VLSTM (M = 1.5429, SE = 0.1253) and it was 1 at the comparison with that of SDLSTM (M = 1.1429, SE = 0.1306). The difference between SDLSTM and VLSTM was marginally significant at $p = 0.074$. Also, the true data has mean of 1.4 and standard error of 0.137

3.6 Summary of the results

In the section of 3.1.1 and 3.1.2, we showed that each neural decoder follows the velocity profile in overall and the reconstructed trajectory reached to the around of the target although the SDLSTM reached to the nearer to the target than that of the other decoders.

Table 1 The summary of the index of velocity prediction performance

Angular difference	CC (x axis)	CC (y axis)	Mean absolute error averaged on x and y axis
VKF < SDLSTM VLSTM < SDLSTM	SDLSTM<VLSTM (marginally significant)	No significant difference	VKF < VLSTM VKF < SDLSTM

The table 1 shows the summary of performance in the velocity prediction. The SDLSTM predicted the movement direction accurately than the other decoders. And the velocity prediction was better than that of VKF and it was similar level with that of VLSTM. At the CC on x axis, the VLSTM was better than the SDLSTM and at the CC on y axis, there were no significant difference between the decoders. However, the CC was not important because it didn't affect actual movement trajectory. It will be discussed in the next chapter.

Table 2 The summary of the index of position prediction performance (part 1)

Euclidean distance with true trajectory	Hit rate	Distance according to time
VKF < VLSTM VKF < SDLSTM	VKF < VLSTM < SDLSTM	VKF < VLSTM VKF < SDLSTM

Also, the performance of the position prediction was reported in the table 2 and table 3. The SDLSTM and the VLSTM predicted the trajectory in higher accuracy than the VKF predicted. Also, the efficiency of approaching to the target was in similar level between VLSMT and SDLSTM and they were better than the VKF. However, because the prediction of the movement direction was best in the SDLSTM, the SDLSTM could acquire the target in highest hit rate (Table 2).

Table 3 The summary of the index of position prediction performance (part 2)

Euclidean distance with ideal trajectory	MDC	ODC
VKF < VLSTM	No significant difference	VLSTM < SDLSTM

The table 3 shows that how smooth the trajectory was (MDC and ODC) and how straightly the trajectory goes to the target (ED with ideal trajectory). The trajectory of the VLSTM was closer to the straight line from the home to target than the trajectory of VKF.

4. Discussion and conclusion

THIS PAGE INTENTIONALLY LEFT BLANK

The aim of this research was to improve the decoding performance by designing new neural decoder based on the characteristics of the data. The literature survey showed that (1) The directional information is explained better with the nonlinear model than with the linear model, (2) There is research implying that the speed and direction is represented differently in the ensemble activity of the motor cortex with the time scale difference or encoding strategy difference. (3) The speed profile has nonlinear characteristics that increase or decrease drastically. If there is difference of the kinematic variables encoding in the motor cortex, it should be decoded differently. Thus, we designed the neural decoder with separate model for predicting speed and movement direction based on the (2). Also, each predictor selected the nonlinear prediction algorithm, LSTM, as the decoding algorithm based on the (1) and (3), which is nonlinearity.

The developed neural decoder, which is speed-direction LSTM (SDLSTM), was compared with the velocity Kalman filter (VKF) and the velocity LSTM (VLSTM). The CRCNS open data was used for training, validating and testing the decoders. We measured the quality of the reconstruction of each decoder in two aspects of accuracy and effectiveness.

Among the accuracy measurement indexes, the SDLSTM showed the poor performance of the correlation coefficient in the velocity prediction at some specific target direction. However, it didn't affect much to the decoded trajectory in two reasons. First, the mean absolute error of the velocity prediction according to the target direction didn't showed any similar pattern with that of the CC graph on the velocity at each axis. For example, the figure 29 showed poor CC on the 90 and -90 directions for all neural decoders. In the other hands, the figure 31 shows that all the decoder's prediction errors were similar between the neural decoders in most of the target direction and didn't showed any specific pattern. In the case of y axis, the pattern of CC graph and the MAE graph were not similar with each other (see figure 30, 32). Thus, the SDLSTM's trajectory could properly reached to the target with highest accuracy of the movement direction prediction and showed highest hit rate regardless of the result of the CC on velocity.

The VKF and VLSTM predicted the speed and the direction together getting the profit from the mutual information between the two variables to predict the variables. The SDLSTM predicted the two variables independently making richer computational complexity could be assigned to each prediction algorithm. Predicting the movement direction independently improved the performance of the movement direction prediction. Thus, it can be thought as that the gain from computational complexity resource was larger than the that from the mutual information. The neural evidence that the variables' encoding method could be different with each other also supports that the weak gain from the mutual information

The number of hidden nodes of the speed predictor in the SDLSTM selected from the random searching

was 13 and that of direction predictor was 36. It implies that the prediction of the direction requires more complexity than that of the speed. Also, the number of epoch and batch sample, which is 76 and 14 in the direction predictor and 57 and 29 in the speed predictor, shows that the prediction of the direction required much more training time and weights update than that of the speed predictor. It implies that the neural encoding complexity in the contralateral primary motor cortex is much higher in the direction variable than in the speed variable because the optimal direction predictor has more complexity than the complexity of the speed predictor.

In conclusion, it was identified that the movement direction could be predicted accurately and target could be acquired with high hit rate by decoding the direction and speed separately with the SDLSTM gaining the profit from the model complexity resource. The velocity reconstruction quality outperformed that of the VKF. Also, in the other accuracy measurement index, the SDLSMT was similar level with the VLSTM and both were better than the VKF. Thus, we identified that the suggested decoder, SDLSTM, could work as the proper neural decoder by improving the overall decoding performance. Further, the results implies that the neural evidence of different encoding strategy of those variables in the motor cortex.

References

THIS PAGE INTENTIONALLY LEFT BLANK

1. Chestek, C. A., Batista, A. P., Santhanam, G., Byron, M. Y., Afshar, A., Cunningham, J. P., ... & Shenoy, K. V. (2007). Single-neuron stability during repeated reaching in macaque premotor cortex. Journal of Neuroscience, 27(40), 10742-10750.

2. Amirikian, B., & Georgopulos, A. P. (2000). Directional tuning profiles of motor cortical cells. *Neuroscience research*, *36*(1), 73-79.

3. Georgopoulos, A. P., Kalaska, J. F., Caminiti, R., & Massey, J. T. (1982). On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. *Journal of Neuroscience*, *2*(11), 1527-1537.

4. Paninski, L., Fellows, M. R., Hatsopoulos, N. G., & Donoghue, J. P. (2004). Spatiotemporal tuning of motor cortical neurons for hand position and velocity. *Journal of neurophysiology*, *91*(1), 515-532.

5. Perel, S., Sadtler, P. T., Godlove, J. M., Ryu, S. I., Wang, W., Batista, A. P., & Chase, S. M. (2013, July). Direction and speed tuning of motor-cortex multi-unit activity and local field potentials during reaching movements. In *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE* (pp. 299-302). IEEE.

6. Li, Z., O'Doherty, J. E., Hanson, T. L., Lebedev, M. A., Henriquez, C. S., & Nicolelis, M. A. (2009). Unscented Kalman filter for brain-machine interfaces. *PloS one*, *4*(7), e6243.

7. Wu, W., Black, M. J., Gao, Y., Serruya, M., Shaikhouni, A., Donoghue, J. P., & Bienenstock, E. (2003). Neural decoding of cursor motion using a Kalman filter. In *Advances in neural information processing systems* (pp. 133-140).

8. Hochreiter, S., urgen Schmidhuber, J., & Elvezia, C. (1997). LONG SHORT-TERM MEMORY. *Neural Computation*, *9*(8), 1735-1780.

9. Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2017). LSTM: A search space odyssey. IEEE transactions on neural networks and learning systems, 28(10), 2222-2232.

10. Karpathy, A. (2016). Cs231n convolutional neural networks for visual recognition. *Neural networks*, *1*.

11. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*(6088), 533.

12. Hinton, G., Srivastava, N., & Swersky, K. (2012). Neural networks for machine learning lecture 6a overview of mini-batch gradient descent.

13. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

14. Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, *13*(Feb), 281-305.

15. Glaser, J. I., Chowdhury, R. H., Perich, M. G., Miller, L. E., & Kording, K. P. (2017). Machine learning for neural decoding. *arXiv preprint arXiv:1708.00909*.

16. Xu, K., Wang, Y., Zhang, S., Zhao, T., Wang, Y., Chen, W., & Zheng, X. (2011, August). Comparisons between linear and nonlinear methods for decoding motor cortical activities of monkey. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE* (pp. 4207-4210). IEEE.