nature human behaviour

Article

https://doi.org/10.1038/s41562-025-02157-x

Human motor cortex encodes complex handwriting through a sequence of stable neural states

Received: 27 June 2024

Accepted: 28 February 2025

Published online: 02 April 2025

Check for updates

Yu Qi[1,2,3,4,7] ✉, Xinyun Zhu[4,7], Xinzhu Xiong[4,7], Xiaomeng Yang[4], Nai Ding[5], Hemmings Wu[6], Kedi Xu[5], Junming Zhu[6] ✉, Jianmin Zhang[6] & Yueming Wang[2,4] ✉

How the human motor cortex (MC) orchestrates sophisticated sequences of fine movements such as handwriting remains a puzzle. Here we investigate this question through Utah array recordings from human MC during attempted handwriting of Chinese characters (n = 306, each consisting of 6.3 ± 2.0 strokes). We find that MC activity evolves through a sequence of states corresponding to the writing of stroke fragments during complicated handwriting. The directional tuning curve of MC neurons remains stable within states, but its gain or preferred direction strongly varies across states. By building models that can automatically infer the neural states and implement state-dependent directional tuning, we can significantly better explain the firing pattern of individual neurons and reconstruct recognizable handwriting trajectories with 69% improvement compared with baseline models. Our findings unveil that skilled and sophisticated movements are encoded through state-specific neural configurations.

Humans are superb at controlling sophisticated fine movements, such as writing, typing and musical performance[1,2]. These sophisticated motoric behaviours are often decomposed into a sequence of simpler movements[3–6]. For example, a word is decomposed into a sequence of letters, a letter can be further decomposed into strokes, and even a stroke may involve a complex movement trajectory. Such decomposition can reduce the complexity and shorten the timescale of each movement unit, which is of particular importance for the motor cortex (MC), since neurons in the MC generally show relatively simple tuning to movement features[7–12], and the tuning varies over relatively long timescales[13]. It remains unclear, however, if primitive units exist during fine movement and how such units may be encoded in the MC.

Handwriting, a skill developed through years of deliberate practice, serves as an example of sophisticated fine motor control in humans. This study investigated the neural basis of handwriting by recording single-unit neural activity from human MC with two 96-channel Utah microelectrode arrays in the left hand knob area of the precentral gyrus[14] (Fig. 1a and Supplementary Fig. 1a). We studied attempted writing of Chinese characters (Fig. 1b,c), which are highly complex (>3,500 frequent characters composed by 32 types of stroke) and pose a great challenge for motor control. We identified that, during the writing of a character, the directional tuning of neurons alternates among a few stable states, each encoding the writing of multiple different small fragments of a character (Fig. 1d–f). Furthermore, the neuronal tuning during handwriting clearly changes across states

[1]Affiliated Mental Health Center and Hangzhou Seventh People's Hospital, MOE Frontier Science Center for Brain Science and Brain-Machine Integration, Zhejiang University, Hangzhou, China. [2]NANHU Brain-Computer Interface Institute, Hangzhou, China. [3]State Key Lab of Brain-Machine Intelligence, Zhejiang University, Hangzhou, China. [4]College of Computer Science, Zhejiang University, Hangzhou, China. [5]Key Laboratory for Biomedical Engineering of Ministry of Education, College of Biomedical Engineering and Instrument Sciences, Zhejiang University, Hangzhou, China. [6]Second Affiliated Hospital of Zhejiang University School of Medicine, Hangzhou, China. [7]These authors contributed equally: Yu Qi, Xinyun Zhu, Xinzhu Xiong. ✉e-mail: qiyu@zju.edu.cn; dr.zhujunming@zju.edu.cn; ymingwang@zju.edu.cn

https://doi.org/10.1038/s41562-025-02157-x

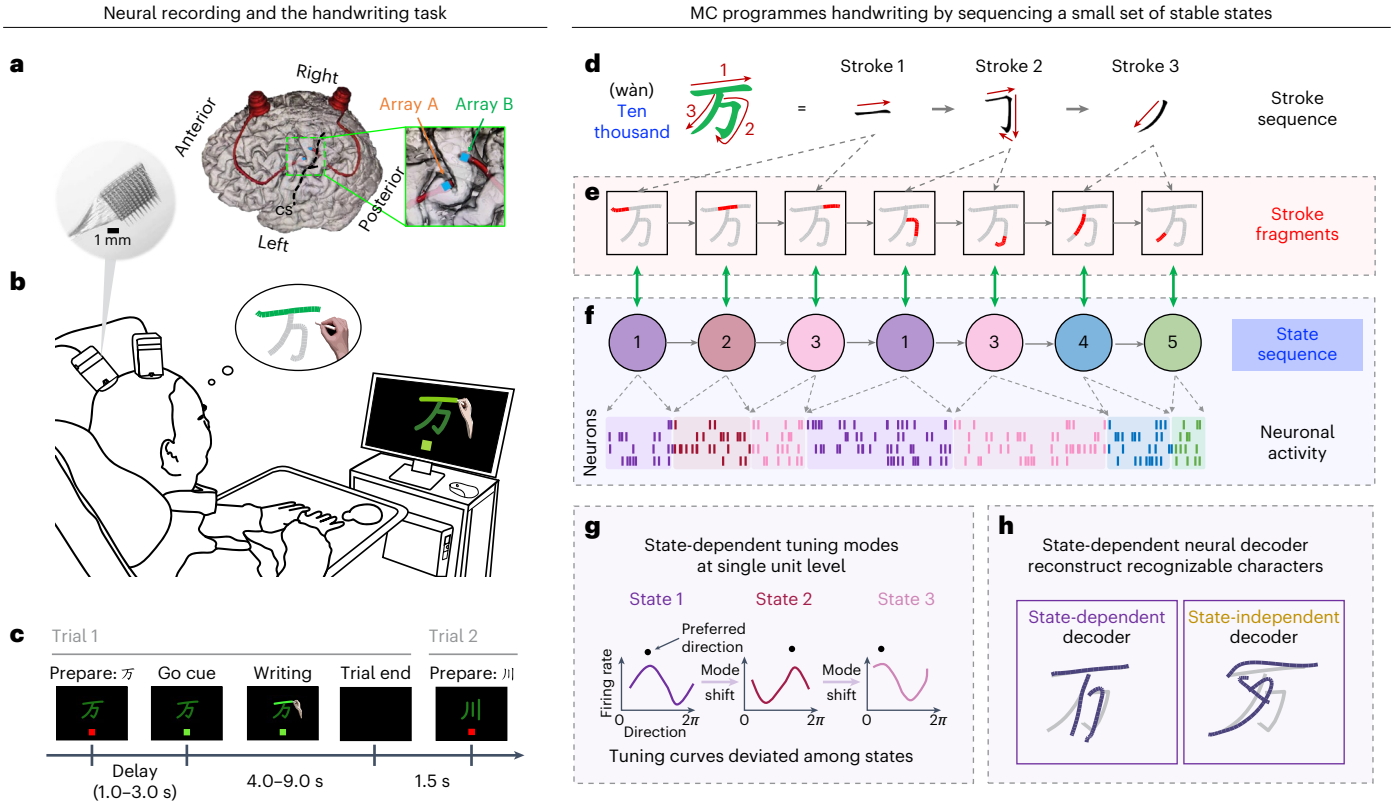


Fig. 1 | The handwriting task and neural representation during attempted handwriting. a, The participant has two 96-channel Utah intracortical microelectrode arrays implanted in the left MC. Both arrays (array A and array B) are positioned in the hand knob area, approximately 1 cm apart from each other. **b**, The participant is instructed to attempt handwriting under video guidance (Supplementary Video 1). **c**, Trial design schematic. Each trial consists of a single character displayed on the screen. A trial begins with the target character being displayed on the screen (prepare), followed by a go cue (green light), and acted writing of the character stroke by stroke (Methods). **d**, Writing of an example Chinese character in terms of strokes (top). **e**–**g**, Summary of the main findings. MC programmes character writing by sequencing a small set of stable states (**f**), each encoding a set of specific stroke fragments (**e**). A set of neurons exhibits directional tuning that is stable within each state but strongly variable across states (**g**). **h**, Writing movements were decoded using both state-dependent (first identifying the neural state and then performing state-dependent decoding for each fragment) and state-independent neural decoders Source data.

(Fig. 1g). Computational models that decompose the writing process into a sequence of states better explain spiking activity from individual neurons (229% change) and better decode the handwriting based on the neural population (69% change; Fig. 1h), compared with a model that assumes stable neuronal tuning throughout the writing of a Chinese character.

Results

Evidence of state-dependent encoding during handwriting

The participant performed attempted handwriting Chinese characters while observing virtual handwriting in the video (Fig. 1a–c, Supplementary Fig. 1a–c and Supplementary Video 1). We recorded 2,850 neurons across 20 experimental sessions (Supplementary Text), and each character was written 3 times. We first analysed whether some neurons reliably responded during the writing of individual characters using the Fisher's discriminant value (FD), which was higher if a neuron generated similar responses when writing the same character twice than when writing two different characters (Fig. 2a and Supplementary Text). A number of neurons showed FD much higher than the chance level, that is, 1.0 ± 0.05 (M ± s.d., estimated by shuffling the responses across characters), demonstrating reliable responses during character writing. We next analysed whether the reliable neural responses to characters could be explained by the classic velocity-based directional tuning model[8,9,15]. The reliability of neural response to individual characters, quantified by the FD, only had a moderate correlation with neural sensitivity to velocity direction, quantified by the R^2 of the directional tuning model (R = 0.46). In other words, neuronal tuning to velocity direction could not well explain neural responses to characters. In the following, to facilitate discussion, we loosely defined two categories of neurons: neurons not well explained by the directional tuning model were referred to as the complex-tuning neurons (n = 115 under the criterion that $R^2 < 0.1$ and FD ≥ 1.4, shown in blue in Fig. 2a), and neurons well explained by the directional tuning model were referred to as simple-tuning neurons (n = 64 under the criterion that $R^2 \geq 0.1$ and FD ≥ 1.4, shown in magenta in Fig. 2a).

To analyse why the directional tuning model failed to perfectly predict the neuronal firing during character writing (as is evidenced by the relatively low R^2 of the model), we illustrated the firing pattern of example neurons by overlaying the neural firing rate during writing on the character trajectory. We observed two general phenomena that were illustrated by two example neurons. First, some neurons reliably responded during the writing of a small fragment of a character, but did not respond during the writing of other fragments that involved movements in the same direction. For example, the same downward vertical movement was involved when writing two characters, that is, '干' and '于', with highly similar orthographical forms but different pronunciations and meanings. A complex-tuning neuron, however, responded differently during the same continuous downward movement in the two characters: it responded to the movement for '于' but did not respond for '干' (Fig. 2b).

Second, some neurons responded to different movement directions in different character fragments. For example, the same neuron may respond to leftward (for example, stroke 1 in '手'), rightward (for example, stroke 2 in '手') and downward (for example, first portion of

https://doi.org/10.1038/s41562-025-02157-x

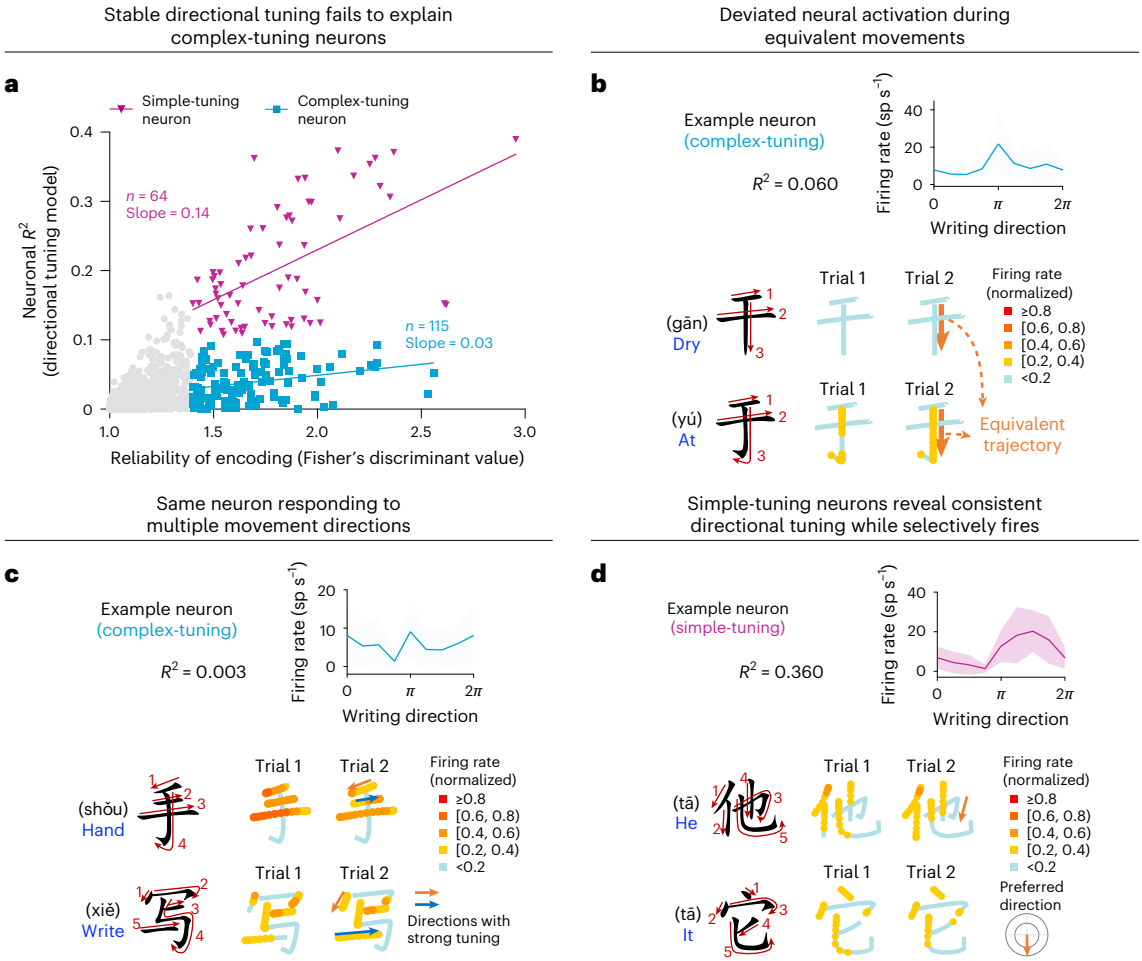


Fig. 2 | Evidence of state-dependent encoding during handwriting. **a**, Reliability and tuning property of neurons (n = 2,850 neurons across 20 sessions). The reliability of the response to each character is characterized using the Fisher's discriminant value (FD) across characters (the 30 within each session), while the tuning property is characterized using the R^2 of the directional tuning model. Neurons with reliable tuning but cannot be explained by the directional tuning model are referred to as complex-tuning neurons (n = 115, R^2 < 0.1 and FD ≥ 1.4), and neurons conformed to the directional tuning model are referred to as simple-tuning neurons (n = 64, R^2 ≥ 0.1 and FD ≥ 1.4). **b**,**c**, Responses of exemplary complex-tuning neurons during handwriting, including the tuning curves (solid line, mean over 90 trials; shaded area, s.d.) and the neural firing rate overlaid on the character trajectory. **b**, An exemplary neuron that shows diverged responses to the same downward movement in two characters. **c**, An exemplary neuron that responds to multiple movement directions but does not consistently respond to any direction. **d**, Responses of exemplary simple-tuning neurons during handwriting, including the tuning curves (solid line, mean over 90 trials; shaded area, s.d.) and the neural firing rate overlaid on the character trajectory. The neuron responds to the downward direction during writing, while it responds to some fragments but not others, although the fragments contain similar trajectories Source data.

stroke 4 in '手') movements when writing some fragments of a character (Fig. 2c), but do not respond to the same movement directions when writing other fragments (for example, first portion of stroke 2 in '写' for rightward movement, first portion of stroke 4 in '手' for downward movement). Even for simple-tuning neurons, we observed that, although strong firing was mainly observed for a certain movement direction, the neuron may or may not fire when the movement direction appears (Fig. 2d). In other words, the neuron was tuned to a movement direction but its response gain seemed to be modulated during the writing process. Therefore, the directional tuning model only explained part of the neuronal activity of simple-tuning neurons (see Supplementary Fig. 2a,b for more examples).

Identification of stable states

Our previous analyses showed that the directional tuning of the neuron might change when writing different fragments. On the basis of these findings, we hypothesized that the writing of a character was decomposed into a sequence of stable states. Motor cortical neurons have stable directional tuning within each state, while they could have different preferred directions in different states.

To identify the stable states, we developed an algorithm, termed temporal functional clustering (TFC; Fig. 3a and Methods), which grouped the movement tuning function in each time bin (50 ms in duration) into a few clusters under a temporal continuity constraint (see Supplementary Fig. 3a,b for verification of the algorithm). A hyperparameter of the model was how many states were allowed. When only one state was allowed, the model was reduced to the classic directional tuning model, and the model complexity increased when the number of states increased.

We found that a small number of states was enough to significantly better predict the neural response than the baseline tuning curve model (P < 0.001 for any state number ≥2, one state versus two states, paired two-tailed t-test, $t(107)$ = 52.36, P < 0.001, Cohen's d = 0.709; Fig. 3b). To determine the optimal number of states, we performed threefold validation for data in each session, where each fold contained 10 distinct Chinese characters that were written 3 times and different folds did not have overlapping characters. The encoding loss on test set converged (decrease of loss <10^{-5}) when about 10 states were modelled (Fig. 3b,c), and the result was consistent across all 18 sessions (Fig. 3d). Bayesian information criterion also indicated that an optimal number of states

https://doi.org/10.1038/s41562-025-02157-x

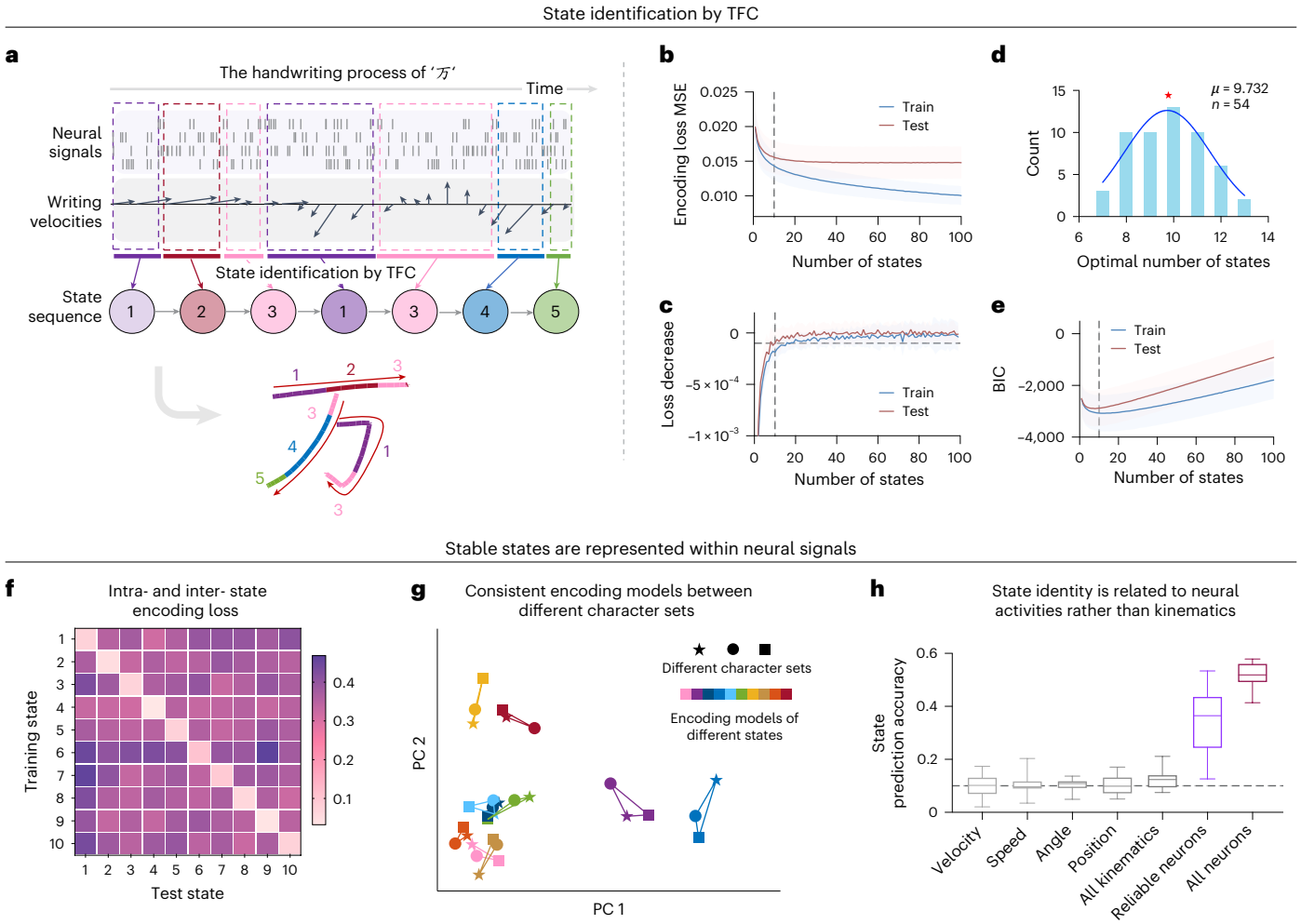


Fig. 3 | Identifying the stable states during handwriting. a, Neural activity recorded during attempted handwriting is divided into non-overlapped stable states, which are classified using the TFC algorithm. Each colour denotes a specific state. **b**,**c**, The model encoding loss decreases when more states are included, and the tipping point occurs around 10 states. A total of 18 sessions, each containing 30 characters, were included (threefold cross-validation, each fold containing 10 characters with 3 repetitions, with non-overlapped characters across folds, M ± s.d.). Both the encoding loss (**b**) and the loss decrease (**c**) are given ($n = 54$, 18 sessions each containing 3 folds). **d**, With a threshold of loss decrease $<10^{-5}$, we found that the mean number of states was around 10 (mean = 9.732, $n = 54$). **e**, Evaluation with the Bayesian information criterion (BIC) also indicated an optimal model number around 10 (the lowest mean BIC was obtained at the state number of 12 and 8 for training and test data, respectively, $n = 54$, M ± s.d.). **f**, Pairwise neural activity encoding loss between states. Each matrix entry (i,j) indicates the prediction error of neural activity in state j for the model trained on state i. A model can only well predict the neural responses during the state it is trained on. **g**, Visualization of encoding models learned by TFC with three non-overlapping character sets. The first two principal components of the linear mapping matrix were plotted. The encoding functions were mostly similar despite different character sets. **h**, State prediction performance of diverse kinematic parameters and neurons. Neural signals can reliably predict the states while kinematic parameters cannot (threefold cross-validation with no overlapping characters, chance level = 0.1). All box plots depict the median (horizontal line inside the box), 25th and 75th percentiles (boxes), and minimum and maximum values (whiskers) Source data.

was around 10 (Fig. 3e). Notably, the number of states used here was much smaller than the number of Chinese characters ($n = 306$) and even the types of stroke for Chinese characters ($n = 32$).

When 10 states were used to model neuronal firing during writing, we found that the directional tuning models learned from one state could well explain neural activity during that state but not other states (Fig. 3f), and the state-dependent tuning models are mostly stable over different character sets (Fig. 3g). In our state-identification process, we clustered the mapping between neural activity and movement directions. Critically, we found that the state information can be also directly decoded from the activation pattern of the neural population but cannot be decoded from kinematic features. These results indicated that the stable states reflect characteristics of the MC, instead of the characteristics of movement features per se (Fig. 3h).

The state-dependent model decomposed the writing process into a sequence of states, with each state encoding multiple fragments that were part of a stroke (stroke fragments, with an average length of 198.4 ± 146.9 ms per fragment; Supplementary Fig. 3c). We observed that a character was decomposed into a similar state sequence in multiple trials (Supplementary Fig. 2c). Consistent results were observed when writing different contents, including English letters, shapes and numbers (Supplementary Fig. 2c). We further found that the state transfer process contained reliable patterns, that is, some states were more likely to appear after others, and was consistent across different character sets (Supplementary Fig. 3d,e). We further observed that the state sequences were associated with specific strokes, regardless of the Chinese character in which the stroke appeared (Supplementary Fig. 3f). This suggests that the state sequence is related to the more abstract meaning of writing, and not just to the movements of writing.

State-dependent tuning of handwriting

The state-dependent model better explained the neural responses of individual neurons than the classic directional tuning model (paired two-tailed t-test, $t(156) = 14.66$, $P < 0.001$, Cohen's $d = 1.265$, threefold

https://doi.org/10.1038/s41562-025-02157-x

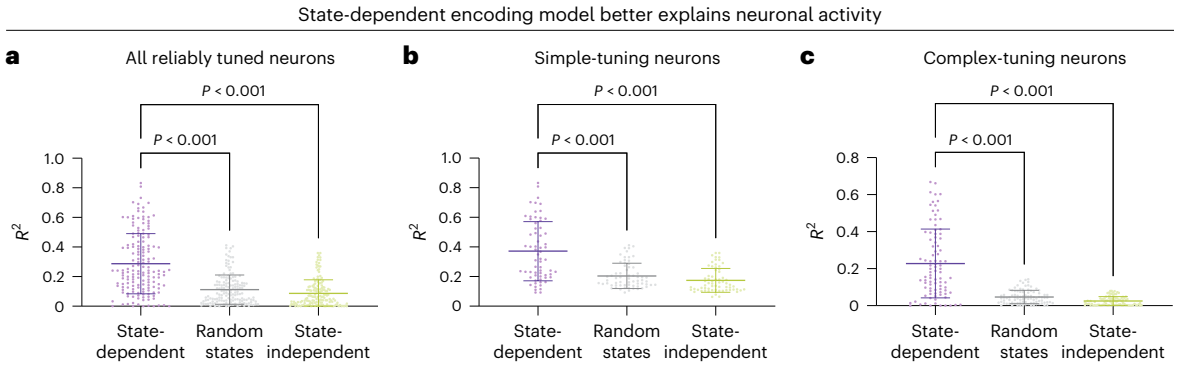


Fig. 4 | State-dependent encoding model better predicts neuronal activities. **a**–**c**, R^2 of the model that explains neural activity using state-dependent directional tuning (with neurons across 20 sessions, M ± s.d.). The state-dependent directional tuning model significantly enhances the R^2 for neurons ($n = 157$) with reliably tuning to characters, and the mean R^2 increased from 0.0873 to 0.2873 (state-dependent versus random states, paired two-tailed t-test, $t(156) = 13.94$, $P = 3.37 \times 10^{-29}$, Cohen's $d = 1.092$; state-dependent versus state-independent, paired two-tailed t-test, $t(156) = 14.66$, $P = 3.897 \times 10^{-31}$, Cohen's $d = 1.265$) (**a**). For simple-tuning neurons ($n = 65$), the mean R^2 increased from 0.1748 to 0.3715 (state-dependent versus random states, paired two-tailed t-test, $t(64) = 8.799$, $P = 1.271 \times 10^{-12}$, Cohen's $d = 1.086$; state-dependent versus state-independent, paired two-tailed t-test, $t(64) = 9.622$, $P = 4.716 \times 10^{-14}$, Cohen's $d = 1.291$) (**b**). For complex-tuning neurons ($n = 92$), the mean R^2 increased from 0.0254 to 0.2279 (state-dependent versus random states, paired two-tailed t-test, $t(91) = 10.77$, $P = 6.226 \times 10^{-18}$, Cohen's $d = 1.353$; state-dependent versus state-independent, paired two-tailed t-test, $t(91) = 11.02$, $P = 1.904 \times 10^{-18}$, Cohen's $d = 1.527$) (**c**) Source data.

cross-validation with no overlapping characters across folds; Fig. 4a), for both simple-tuning neurons (Fig. 4b) and complex-tuning neurons (Fig. 4c). For complex-tuning neurons, the mean R^2 increased from 0.03 to 0.23 (paired two-tailed t-test, $t(91) = 11.02$, $P < 0.001$, Cohen's $d = 1.527$, 95% confidence interval for the difference in mean R^2 (0.17, 0.24)). The state-dependent model also better explained the neuronal activities when writing English letters, shapes and numbers (Supplementary Fig. 4). In the following, we explored why the state-dependent model improved the modelling of simple- and complex-tuning neurons.

We first examine the directional tuning behaviour of neurons across states. For each neuron, the state-dependent model estimated a directional tuning curve per state, and we illustrated these tuning curves for both simple- and complex-tuning models (Fig. 5a for normalized neural responses against directions of all neurons and Fig. 5b,c for tuning curves of representative neurons). For simple-tuning but not complex-tuning neurons, the tuning had similar preferred directions across different states (Fig. 5a,b). To quantify this effect, we analysed the preferred direction of each tuning curve (PD, representing the direction with the largest firing rates) and the standard deviation of PD was higher for complex-tuning than simple-tuning neurons (unpaired two-tailed Kolmogorov–Smirnov test, $D(64,115) = 0.628$, $P < 0.001$; Fig. 5d). Next, we further analysed the modulation depth of each tuning curve (MD, defined as the peak-to-peak value of the directional tuning curve). It was found that the MD varied across states even for simple neurons and the standard deviation of MD was comparable for complex- and simple-tuning neurons (unpaired two-tailed Kolmogorov–Smirnov test, $D(64,115) = 0.200$, $P = 0.063$; Fig. 5e). The state-dependent MD could explain why the state-dependent model outperformed the state-independent model even for simple-tuning neurons.

Handwriting decoding based on stable states

The previous analyses demonstrated how individual neurons encode the handwriting process, and in the following, we utilized the population response to decode the handwriting trajectory. We built a state-dependent neural decoder to decode the movement velocity vector (direction with speed) and recovered the trajectory of each stroke based on velocity (Fig. 6a). To enable state-dependent decoding, we extended the velocity Kalman filter with a dynamic observation function (the linear mapping from kinematics to neural signals)[14]. Specifically, instead of using a static observation function, the state-dependent decoder contained a pool of observation models (which are the linear directional tuning models learned by TFC), and adaptively weighed and assembled the models based on the Bayesian inference of states, given the incoming neural signals.

The state inferred by the state-dependent neural decoder was generally consistent with the state inferred by the neural encoding model (Supplementary Fig. 5a,b). Compared with a state-independent neural decoder (velocity Kalman filter[9]), the state-dependent decoder could decode handwriting with lower root mean squared error (RMSE; 13–18% decrease, paired two-tailed t-test, $t(17) = 31.20$, $P < 0.001$, Cohen's $d = 2.199$; Fig. 6b,c) and higher R^2 (>69% increase, paired two-tailed t-test, $t(17) = 31.37$, $P < 0.001$, Cohen's $d = 2.495$, Fig. 6b,d, threefold cross-validation with no overlapping characters across folds). Consistent results were obtained with both single-unit neural activity (SUA, offline sorted) and multi-unit neural activity (MUA, unsorted; Supplementary Fig. 5c). Significant improvement was also achieved when writing different contents, including English letters, shapes and numbers (Supplementary Fig. 6). Overall, these results confirmed that the state-dependent approach enhanced the decoding of stroke fragment trajectories during handwriting.

The decoding process could be achieved online, allowing the participant to use a robotic arm to write recognizable Chinese characters through attempted writing (Supplementary Video 2; the strokes were decoded by the state-dependent decoder, while the end of the strokes and the beginning of the next strokes were predefined to reconstruct the character from the strokes). We further evaluated the long-term reliability of handwriting decoding and found that the same state-dependent decoder could perform stably for a month (Supplementary Fig. 5d). Furthermore, its performance generalized to new characters (Supplementary Fig. 5d). Taken together, these results demonstrated that complex handwriting is decomposed into smaller units that are encoded in different neural states in the MC, and only within each state is movement encoded in a roughly linear manner.

Discussion

Handwriting is a highly sophisticated skilled behaviour that is special to the human being, and the inherent complexity of the Chinese writing task provides a unique window to dissect the underlying neural mechanisms for sophisticated fine movements. Through the task, we found that the human MC encodes a sophisticated writing process with a sequence of stable states, involving two types of neuron:

https://doi.org/10.1038/s41562-025-02157-x

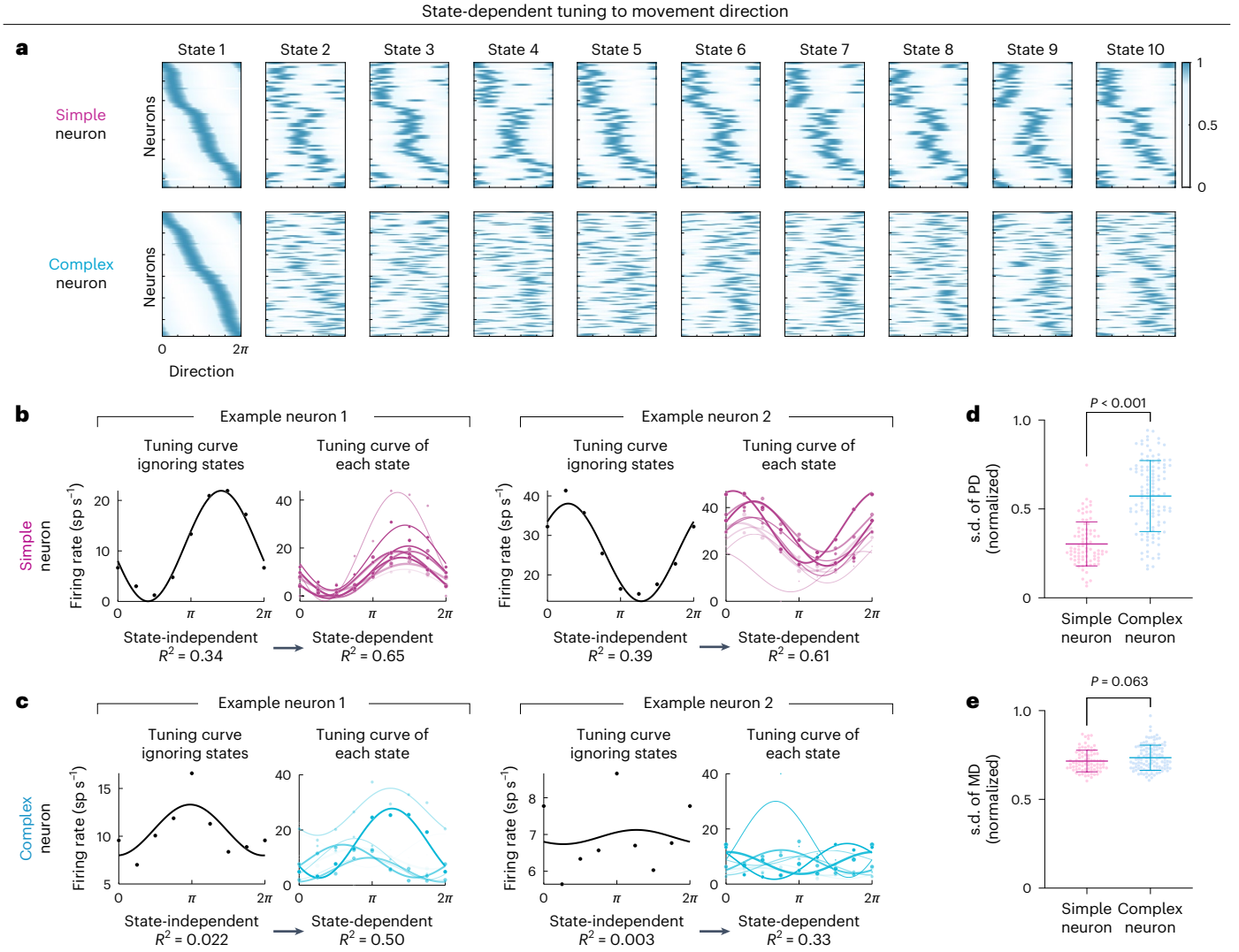


Fig. 5 | State-dependent directional tuning of MC neurons. a, Normalized firing rates over directions with both simple- and complex-tuning neurons, sorted based on state 1. The simple-tuning neurons show more consistent directional tuning over states compared with the complex-tuning neurons. **b**, Directional tuning curve of two exemplar simple-tuning neurons estimated using a state-independent model (black) or a state-dependent model (magenta). For the state-dependent model, a tuning curve is estimated based on each state and the tuning curves have a consistent preferred direction (PD) across states but the modulation depth (MD) varies across states. Darker colour for curves with higher R^2. **c**, Similar to **b**, but for complex-tuning neurons that exhibit variable directional tuning under different states. **d**,**e**, Statistical analysis of PD (**d**) and MD (**e**) for simple- and complex-tuning neurons. The standard deviation of PD is significantly higher for complex-tuning neurons (mean = 0.57, $n = 115$) than simple-tuning neurons (mean = 0.30, $n = 64$) (unpaired two-tailed Kolmogorov–Smirnov test, $D(64,115) = 0.628$, $P = 1.97 \times 10^{-20}$ (**d**); unpaired two-tailed Kolmogorov–Smirnov test, $D(64,115) = 0.200$, $P = 0.063$ (**e**)) Source data.

(1) simple-tuning neurons that have stable directional tuning and (2) complex-tuning neurons that have variable directional tuning across states.

These results support a hierarchical control scheme of sophisticated movements (Supplementary Fig. 7): hypothetically, a highly complex movement trajectory is first decomposed into small and simple fragments, and each trajectory fragment is further converted into a sequence of movement velocities under a specific state of the MC. Furthermore, in this process, it is possible that (1) the complex-tuning neurons are upstream motor neurons related to the encoding of trajectory fragments, and (2) the simple-tuning neurons are downstream motor neurons related to the encoding of movement velocity. In other words, the complex-tuning neurons decompose a trajectory fragment into momentary velocity, and the movement velocity is implemented by the simple-tuning neurons. This two-step movement control perspective is in sharp contrast with the traditional perspective that the MC only carries out simple motor commands[7,16,17].

Hierarchically decomposing a complex sequence into smaller units is a common strategy in the brain[5] and its neural underpinnings have been studied across multiple domains such as motor planning[3], decision-making[18], working memory[19], song sequencing[20] and language processing[21]. Revealing the primitive units to decompose a sequence, however, turns out to be highly challenging, especially for a language-related process[6,22]. On the basis of the writing behaviour, where the movements are associated with abstract meanings, characters and strokes are apparent candidates for the primitive unit for writing. Nevertheless, Chinese has a large number of characters (>3,500 frequent characters), and assigning a specific neural population for the writing of each character, that is, one-hot coding, is implausible. The number of stroke types is limited ($n = 32$). Nevertheless, strokes can be sophisticated, such as '乙' and '马', and redundant, that is, sharing common elements such as vertical or horizontal movements. Furthermore, even the writing of a simple stroke such as '—' can be meaningfully divided into a start, a body and an end, which

https://doi.org/10.1038/s41562-025-02157-x

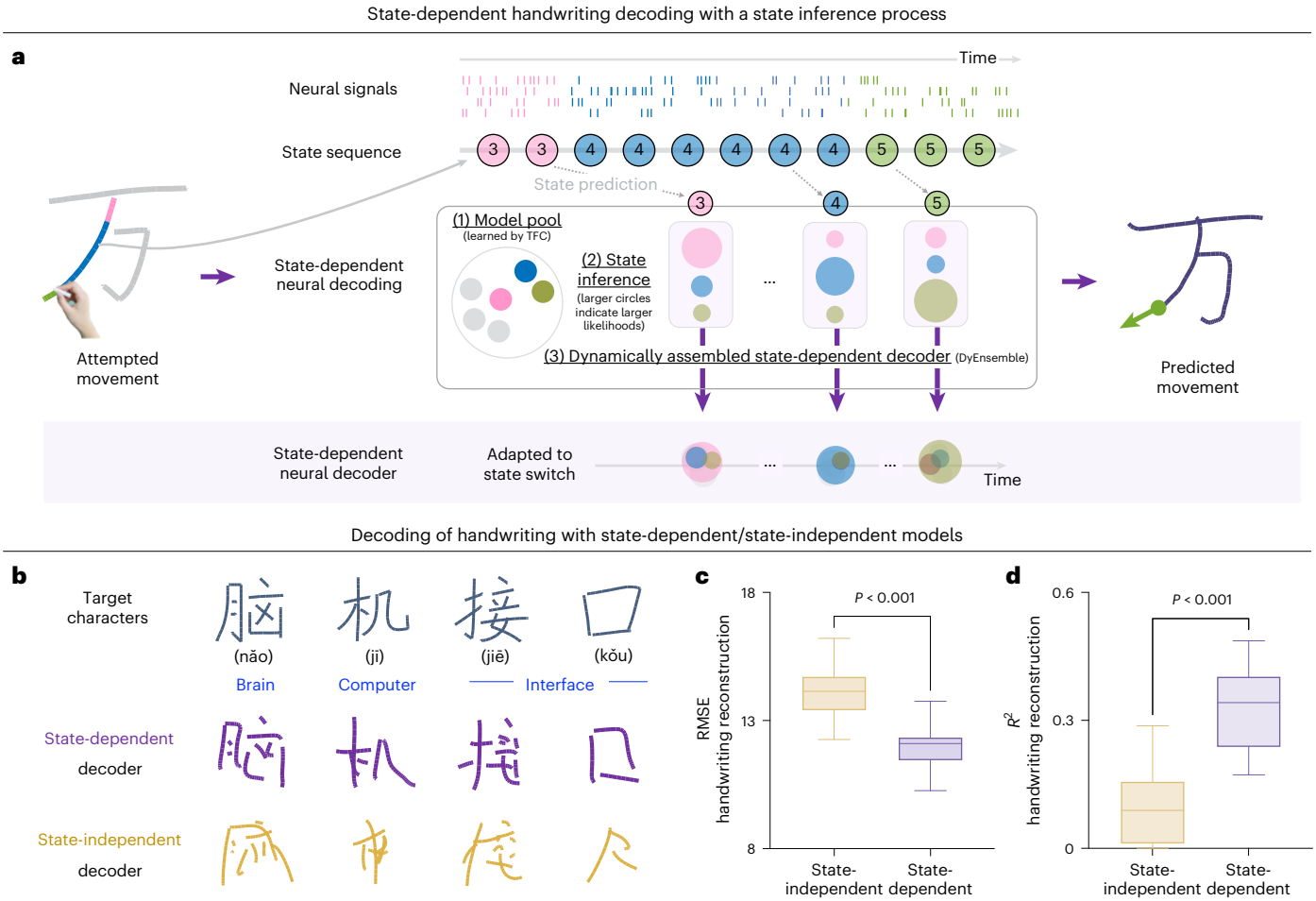


Fig. 6 | State-dependent decoding model improves the performance of handwriting trajectory prediction. a, Diagrammatic representation of the state-dependent decoding process during handwriting (DyEnsemble). Utilizing encoding models for each state established through TFC, DyEnsemble dynamically infers the state and adaptively assembles a state-specific decoder in real time based on the incoming neural signals. This approach allows for adaptive switching between decoding models along with state switches. **b**, The writing trajectory was decoded using state-dependent and state-independent neural decoders. **c**,**d**, Performance of handwriting decoding using RMSE and R^2 between the ground truth trajectory and the decoded trajectory (offline evaluation with threefold cross-validation with no overlapping characters across folds in each session, $n = 18$; paired two-tailed t-test, $t(17) = 31.20$, $P = 1.896 \times 10^{-16}$, Cohen's $d = 2.199$ (**c**); paired two-tailed t-test, $t(17) = 31.37$, $P = 1.734 \times 10^{-16}$, Cohen's $d = 2.495$ (**d**)). All box plots depict the median (horizontal line inside the box), 25th and 75th percentiles (boxes), and minimum and maximum values (whiskers) Source data.

are especially emphasized in Chinese calligraphy and correspond to the decomposition of a movement into acceleration and deceleration phases[23]. Here we demonstrate that, in the MC, the writing of a character is decomposed into a sequence of states that correspond to fragments of a stroke. We also found that the state sequence was related to the strokes being written, which is the abstract meaning of writing rather than the movement parameters. This suggested that cognitive information was involved during handwriting encoding: state signals may originate from cognitive processes, while the MC receives the signals and performs state-dependent tuning during handwriting control.

Previous studies have demonstrated that neurons in the MC are tuned to movement features[7,8,11,24–26] but static tuning to movement features has limited ability to explain the variance of neuronal activity[13,23,27], especially during natural movements[28]. Instead, neural tuning to movement features can actively adapt to a specific task or environmental setting[29,30]. In other words, depending on the external environment or instruction, the MC neurons can encode movements through different neural encoding subspaces[31–33]. The current study, however, demonstrates intrinsic alternation between MC neural encoding subspace, depending on internally generated states during sophisticated movements. Similar ideas have also been proposed based on relatively simple movements, for example, decomposing a reaching movement into an acceleration phase and a deceleration phase[13,23,34,35], stereotyped movement fragments[12], but here we demonstrated many diverse states, which can encode various movement fragments using under a linear directional tuning model, and suggested a hierarchical model for the neural control of sophisticated movements. We further found that state transition occurred at the neural population level, supporting recent findings on population-level dynamics in MC[36,37].

A limitation of this study is that the results shown are from only one participant, and an important next step is to validate the results with more individuals. Second, in the experimental paradigm, the participant not only performed attempted handwriting but was simultaneously observing the virtual handwriting of the character. Thus, the neural signals could also contain the modulation of the observation process[38,39]. Third, the proof-of-concept demonstration is still an open-loop control as the participant was watching the virtual hand in the video instead of the robotic arm writing the strokes (Supplementary Video 2). Further advances in decoders and systems are required to enable high-performance handwriting BCIs in a closed-loop manner. In addition, previous studies also identified primitive units in movements based on kinematic parameters[40–43] or muscle activities[44–48], indicating a hierarchical movement segmentation at different levels.

The relationship between segmentations with different signals can be valuable for future work.

In summary, our results strongly demonstrate that sophisticated fine movements such as handwriting are encoded in the human MC as a sequence of stable states, each encoding a fragment of the movement, and the state-dependent encoding mechanism can shed light on future design of brain-computer interfaces for sophisticated fine movements.

Methods

Experimental model and subject details

Participant and ethics. All clinical and experimental procedures conducted in this study received approval from the Medical Ethics Committee of The Second Affiliated Hospital of Zhejiang University (ethical review number 2019-158, approved on 22 May 2019) and were registered in the Chinese Clinical Trial Registry (ref. ChiCTR2100050705). Informed consent was obtained verbally from the participant, along with the consent of his family members, and was duly signed by his legal representative. This study used an observational design and no intervention took place that was not driven by clinical need.

The volunteer participant is a right-handed man, 75 years old at the time of data collection. He was involved in a car accident and suffered from complete tetraplegia subsequent to a traumatic cervical spine injury at the C4 level, which occurred approximately 2 years before study enrolment. The volunteer participant demonstrated the ability to move body parts above the neck and exhibited normal linguistic competence and comprehension for all tasks. He scored 0/5 on skeletal muscle strength for limb motor behaviour.

On 27 August 2019, two 96-channel intracortical microelectrode arrays (4 mm × 4 mm Utah array with 1.5 mm length, Blackrock Microsystems) were implanted in the left MC, with one array located in the middle of the hand knob area (array A) and the other array located medially approximately 1 cm apart (array B), guided by structural (CT) and functional imaging (fMRI)[49]. The participant was asked to perform imagery movement of hand grasping and elbow flexion/extension with fMRI scanning to confirm the activation area of the MC[49]. Data presented in this study cover the period from post-implant days 1,374 to 1,792.

Method details

Neural signal recording and processing. Neural signals were recorded from the microelectrode arrays using the Neuroport system (NSP, Blackrock Microsystems). The signals were amplified, digitized and recorded at a sampling rate of 30 kHz. To reduce common mode noise, a common average reference filter was applied, subtracting the average signal across the array from each electrode. A digital high-pass filter with a cut-off frequency of 250 Hz was then applied to each electrode. Then, threshold crossing detection was performed using the Central software suite (Blackrock Microsystems). The threshold was set based on the root mean square (RMS) of the voltage time series recorded on each electrode. Specifically, thresholds ranging from −6.25 × RMS to −5.5 × RMS were used.

To analyse the neuronal activity, neurons were manually sorted with either Plexon Offline Spike Sorter v4 (offline analysis) or Central software (online decoding). After spike sorting, neuronal spikes were binned into 50 ms bins, without overlapping (each stroke contains 12 ± 6 bins, and each character contains 126 ± 24 bins). For the decoding tasks, the spike data were smoothed using an average filter of 5 bins.

The delay between the start of data extraction and the go cue was determined by evaluating a set of delay values ranging from −1,000 ms to 1,000 ms. The delay value of 300 ms was selected according to the overall decoding performance with a linear model.

Experimental paradigm. The participant performed an attempted handwriting task while observing a virtual handwriting of the character. During the experimental sessions, the participant attempted the handwriting of characters guided by instructional videos (Fig. 1 and Supplementary Video 1). The instructional video presented a virtual hand writing a target character stroke by stroke; at the same time, the participant attempted writing the character as if the virtual hand belonged to him.

For the Chinese character writing task, there were two paradigms.

1. Single-character writing with visual guidance (CW): a single Chinese character in dark green colour was displayed on the screen above a red square during the delay period, which lasted between 1 s and 3 s. After the delay period, the red square cue turned green and played a sound 'ding' to signal the participant to start writing. Simultaneously, a virtual hand holding chalk appeared on the screen and wrote the character stroke by stroke, highlighting the written part in bright green. The duration of the writing period varied depending on the complexity of the Chinese character, with more strokes requiring a longer writing time. After completing one character, the screen turned black and lasted for 1.5 s (Supplementary Video 1).
2. Sentence writing with visual guidance (SW): initially, the characters of a sentence were displayed in dark green on the screen above a red square. Following a delay period, the red square turned green, indicating the start of the sentence. The experimental procedure for each character within the sentence was identical to the visual guidance paradigm, with a short horizontal line appearing below the character as a reminder for the participant to begin writing a character (Supplementary Video 2).

The instructional video presents the writing process in a stroke-by-stroke manner. To facilitate smooth tracking, the velocity of the virtual hand was programmed to maintain a constant acceleration, ensuring ease of following for the participant.

The 'single-character writing with visual guidance' paradigm was used for data collection and offline analysis. The 'sentence writing with visual guidance' was mostly for online evaluation and demonstration. The main results were analysed offline, and the signal processing and analysis systems were developed by MATLAB and Python. For the online evaluation, the neural signal processing programme was developed in MATLAB, and the graphical user interface of the experimental task was developed in Python (see Supplementary Text for online settings).

TFC algorithm. The TFC algorithm is a computational approach designed to identify stable state and state switches during writing. The underlying assumption of TFC is that the neuronal encoding model, at the population level, may shift under different states while keeping stable within the same state. Under this assumption, TFC can automatically compute the neural mapping model for each state and detect state switches in a data-driven way.

Given a paired dataset $\{X, Y\} \equiv \{x_1, \ldots, x_T, y_1, \ldots, y_T\}$, where $X \in \mathbb{R}^{d_x \times T}$ represents the writing kinematic data, and $Y \in \mathbb{R}^{d_y \times T}$ represents the preprocessed neural data, with d_x and d_y denoting the respective dimensions, and T representing the length of time bins. Our objective is to learn M encoding models of $\mathcal{H}_m(\bullet) \in \{\mathcal{H}_1(\bullet), \mathcal{H}_2(\bullet), \ldots, \mathcal{H}_M(\bullet)\}$, where each model represents an encoding in a single state. In this study, each encoding model is a linear mapping from the velocity vector to the neural signals, that is, the velocity-based directional tuning model.

To obtain the M encoding models, TFC first randomly selected a set of encoding models and then applied the models to each data pair. The data would be assigned to the encoding model with minimal errors, and the models would update with the data assigned to them. The process repeated until the errors were minimized.

Specifically, the TFC algorithm uses an Expectation-Maximum (EM) process. Initially, we randomly set the parameters of M encoding models. Then we perform the Expectation-step (E-step), where each pair of $\{X,Y\}$ is assigned to a specific encoding model where the data has the lowest encoding loss. Especially, the encoding loss is smoothed

temporally to encourage that the adjacent data pair should belong to the same state (temporal constraint). After that, we perform the Maximum-step (M-step), where each model updates its parameters with the data pairs assigned to it. The TFC algorithm repeats the E-step and M-step iteratively until convergence.

1. Initialization step
   To obtain M initial models, we randomly set the parameters of each model $\mathcal{H}_m(\bullet)$. Each model is an encoding function that maps kinematics to neural signal estimation $\hat{Y}$:

$$\hat{Y} = \mathcal{H}_m(X). \tag{1}$$

2. E-step: data assignment
   The E-step aims to reassign data pairs to their fittest models. For each data pair $\{X, Y\}$, we can compute the predicted neural signal $\hat{Y}_m$ given kinematics X and encoding model $\mathcal{H}_m(\cdot)$:

$$\hat{Y}_m = \mathcal{H}_m(X) \tag{2}$$

where $\hat{Y}_m \in \mathbb{R}^{d_y \times T}$ denotes the neural signals predicted by the encoding model $\mathcal{H}_m(\cdot)$ for all the kinematics X. Given a set of M encoding models, we can obtain M predictions for each data pair. Then we calculate the encoding loss at each time step for each encoding model:

$$E_m = \sum \sqrt{||Y - \hat{Y}_m||^2} \tag{3}$$

where $E_m \in \mathbb{R}^{1 \times T}$ is the encoding error between the neural signal $\hat{Y}_m$ estimated by model $\mathcal{H}_m(\cdot)$, and the ground truth Y, summed over all the neurons. Next, we smooth the error vector E_m temporally to encourage data pairs that are adjacent in time to be assigned to the same model. With this constraint, we can obtain more temporally continuous state segmentation. The window size for a moving averaging smooth was typically set with 3 to 10 bins, denoted as l_{smooth}. The smoothed encoding loss for model $\mathcal{H}_m(\cdot)$ is given by

$$E_m = \text{smooth}(E_m, l_{\text{smooth}}). \tag{4}$$

Practically, the parameter of l_{smooth} could affect the length of states, and a larger l_{smooth} usually leads to bigger lengths for states. Here we statistically analysed the length of states with different settings of l_{smooth}. As shown in Supplementary Fig. 3c, the mean length of states was mostly around 200 ms across different settings. Once we have the smoothed encoding loss for each model, we assign each data pair to the model that gives the minimal encoding loss:

$$E = [E_1; E_2; \ldots; E_m; \ldots; E_M], \tag{5}$$

$$m_{\text{assign}} = \text{argmin}_m(E) \tag{6}$$

where $E \in \mathbb{R}^{M \times T}$ represents each model's encoding loss and $m_{\text{assign}} \in \mathbb{R}^{1 \times T}$ denotes the model index selected.

3. M-step: parameter updating
   After the E-step, each data pair has been assigned to a specific model, and each model $\mathcal{H}_m(\cdot)$ has a collection of data pairs $\{X_{\mathcal{H}_m}, Y_{\mathcal{H}_m}\}$. Then we can update the parameters for the models, by fitting the following function:

$$Y_{\mathcal{H}_m} = \mathcal{H}_m(X_{\mathcal{H}_m}) + \epsilon_{\mathcal{H}_m} \tag{7}$$

with $\epsilon_{\mathcal{H}_m}$ being the zero-Gaussian noise term. In this study, we use a linear function, such that $\mathcal{H}_m(\cdot)$ would include a linear mapping matrix and a Gaussian noise.

4. Repeat E-step and M-step until convergence

By iteratively performing the E-step and M-step, the overall encoding error will decrease continuously. Suppose at the ith iteration we have an error of e_i. One pre-set early-stopping threshold β (typically set to 0.001) can be used to decide when to stop the iteration:

$$e_i < \beta e_{i-1}. \tag{8}$$

When the iteration stops, the TFC algorithm will return the parameters for each model (that is, the directional tuning model) as well as the temporal segments corresponding to each model. The temporal segments indicate the segmentation of states.

Existing approaches, such as the hidden Markov model (HMM), can also segment movements into states[23]. However, the HMM model only inputs neural signals, such that the temporal bins with similar neural signal patterns will be regarded as the same state. Meanwhile, TFC defines neural states as stable directional tuning between neural activities and movements, where temporal bins with similar tuning functions are regarded as the same state, and thus facilitate the state-dependent directional tuning model.

State-dependent neural decoder. A state-dependent neural decoder was proposed to allow dynamic switching of decoding models adaptively with changes in states (Fig. 6a). The state-space model of the Kalman filter includes two parts: a system function (mapping from previous kinematics to current kinematics) and an observation function (mapping from kinematics to neural signals). To enable state-dependent decoding, we extended the Kalman filter with a dynamic observation function, in a dynamic ensemble framework (DyEnsemble)[14]. The DyEnsemble uses the encoding models coming from the TFC algorithm as a pool of observation models and adaptively weighs and assembles the models based on the Bayesian rule, given the incoming neural signals. In this way, the DyEnsemble enables state-specific decoding that adapts to the change of states[14,50].

The DyEnsemble decoder defines the state-space model as follows:

$$x_k = f(x_{k-1}) + \varsigma_{k-1}, \tag{9}$$

$$y_k = h_k(x_k) + \varepsilon_{v_k}. \tag{10}$$

The model contains a system equation (equation (9)) that transmits the value of interest x_{k-1} at time $k-1$ to the next time step k using function $f(\bullet)$, with $\varsigma_k \sim N(0, \sigma_\varsigma^2)$, a zero-mean Gaussian, being the transition noise. It also contains an observation equation (equation (10)), which uses x_k, to infer the measurement variable y_k using function $h_k(\bullet)$, with a zero-mean Gaussian observation noise ε_{v_k}. Notably, the observation function $h_k(\bullet)$ is dependent on time, such that it dynamically changes over time. The DyEnsemble algorithm aims to estimate the dynamic measurement function ($h_k(\bullet)$), and the corresponding x_k over time, given the incoming neural signals (y_k).

In the scenario of neural decoding, $x_k \in \mathbb{R}^{d_x}$ is the movement kinematics, and $y_k \in \mathbb{R}^{d_y}$ denotes the neural signals, with d_x and d_y denoting the respective dimensions. The observation function $h_k(\bullet)$ is an encoding function mapping from kinematics (x_k) to neural signals (y_k). DyEnsemble model maintains a pool of encoding models as $\{\mathcal{H}_1(\bullet), \mathcal{H}_2(\bullet), \ldots, \mathcal{H}_M(\bullet)\}$. Given incoming neural signals y_k, DyEnsemble weighs the models in the pool by the Bayesian likelihood to y_k and assembles $h_k(\bullet)$ in a Bayesian averaging rule. Here we have obtained the model pool with the TFC algorithm, with each model representing a specific state. Therefore, the DyEnsemble model can facilitate state-specific decoding by first inferring the state identity and then adaptively switching to the proper model.

Note that, if we consider a one-state condition, where the observation equation (equation (10)) uses a constant model instead of a state-dependent $h_k(\bullet)$, the equations can be solved with the Kalman filters for an optimal solution. Therefore, we used the Kalman filter as the state-independent decoder for comparison.

Specifically, considering a time series of neural signals $y_{0:k}$, the decoding problem involves estimating the kinematic state x_k at time step k. The posterior distribution of the state can be specified by

$$p(x_k,|,y_{0:k}) = \sum_{m=1}^{M} p(x_k,|,h_k(\cdot)=\mathcal{H}_m(\bullet),y_{0:k})\,p(h_k(\cdot)=\mathcal{H}_m(\bullet),|,y_{0:k}). \tag{11}$$

Here $p(x_k,|,h_k(\cdot)=\mathcal{H}_m(\bullet),y_{0:k})$ represents the kinematic state posterior estimated by the model $\mathcal{H}_m(\bullet)$ at time k, and $p(h_k(\cdot)=\mathcal{H}_m(\bullet),|,y_{0:k})$ denotes the posterior probability of selecting the encoding model $\mathcal{H}_m(\bullet)$ at time k, which can be computed as follows:

$$p(h_k(\cdot)=\mathcal{H}_m(\bullet),|,y_{0:k}) = \frac{p(h_k(\cdot)=\mathcal{H}_m(\bullet),|,y_{0:k-1})\,p_m(y_k,|,y_{0:k-1})}{\sum_{j=1}^{M} p(h_k(\cdot)=\mathcal{H}_j(\bullet),|,y_{0:k-1})\,p_j(y_k\,|\,y_{0:k-1})} \tag{12}$$

where $p(h_k(\cdot)=\mathcal{H}_m(\bullet),|,y_{0:k-1})$ represents the prior probability of choosing model $\mathcal{H}_m(\bullet)$ at time k, while $p_m(y_k\,|\,y_{0:k-1})$ is the marginal likelihood of choosing model $\mathcal{H}_m(\bullet)$ at time k. The marginal likelihood represents the confidence or reliability of a particular model's prediction.

The prior probability at time k can be recursively expressed as the posterior probability at time $k-1$, with a sticking factor α:

$$p(h_k(\cdot)=\mathcal{H}_m(\bullet)\,|\,y_{0:k-1}) = \frac{\left[p(h_{k-1}(\cdot)=\mathcal{H}_m(\bullet),|,y_{0:k-1})\right]^{\alpha}}{\sum_{j=1}^{M}\left[p(h_{k-1}(\cdot)=\mathcal{H}_j(\bullet),|,y_{0:k-1})\right]^{2}} \tag{13}$$

where $p(h_{k-1}(\cdot)=\mathcal{H}_m(\bullet),|,y_{0:k-1})$ is the posterior probability of choosing encoding model $\mathcal{H}_m(\bullet)$ at time $k-1$. The parameter $\alpha \in (0,1)$ represents the sticking factor, where a higher value leads to smoother changes in the model weights.

And the marginal likelihood can be computed as follows:

$$p_m(y_k,|,y_{0:k-1}) = \int p_m(y_k,|,x_k)\,p(x_k,|,y_{0:k-1})\,dx_k \tag{14}$$

where $p_m(y_k,|,x_k)$ is the likelihood of model $\mathcal{H}_m(\bullet)$ for a specific kinematic state x_k. The DyEnsemble model can be solved using a particle filtering algorithm (see Supplementary Text for details).

Statistical analysis

In Fig. 3b, we applied the paired t-test (two-tailed), the $P < 4.417 \times 10^{-78}$ for any state number ≥2. In Fig. 4a–c, we applied the paired t-test (two-tailed), with $n = 157$, $n = 65$ and $n = 92$. The P values are 3.37×10^{-29}/3.897×10^{-31} (state-dependent versus random states/state-dependent versus state-independent), 1.271×10^{-12}/4.716×10^{-14} and 6.226×10^{-18}/1.904×10^{-18}, respectively. In Fig. 5d,e, we applied the Kolmogorov–Smirnov test (two-tailed). For Fig. 5d, $n = 64$ and 115, with $P = 1.97 \times 10^{-20}$. For Fig. 5e, $n = 64$ and 115, with $P = 0.06299$. In Fig. 6c,d, we applied the paired t-test (two-tailed), with $n = 18$ sessions. The P values are 1.896×10^{-16} and 1.734×10^{-16} for the RMSE and R^2, respectively. In Supplementary Fig. 3e, we applied a paired t-test (two-tailed) with $n = 18$ sessions. The P value is 1.373×10^{-12}. In Supplementary Fig. 4, we applied the paired t-test (two-tailed) with $n = 157$ reliable neurons. The P values for English letters, shapes and numbers are 2.098×10^{-7}/1.642×10^{-8} (state-dependent versus random states/state-dependent versus state-independent), 2.556×10^{-5}/1.133×10^{-6} and 5.77×10^{-7}/5.865×10^{-9}, respectively. In Supplementary Fig. 5c, we applied a paired t-test, with $n = 18$ sessions. The P values are 3.49×10^{-4} and 6.507×10^{-8} for state-independent and state-dependent. In Supplementary Fig. 6b, we applied a paired t-test (two-tailed). The P values are 3.385×10^{-14}, 3.229×10^{-12}, 1.567×10^{-12} and 6.614×10^{-10} for the characters, English letters, shapes and numbers, respectively.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

Data relevant to this study are accessible under restricted access according to our clinical trial protocol. Access can be granted upon request to the corresponding author. The response can be expected within 3 weeks. Any data provided must be kept confidential and must not be shared with others without approval. Source data are provided with this paper.

Code availability

The code used for analyses in this paper is available via Zenodo at https://zenodo.org/records/14865736 (ref. 51).

References

1. Shenoy, K. V., Sahani, M. & Churchland, M. M. Cortical control of arm movements: a dynamical systems perspective. *Annu. Rev. Neurosci.* **36**, 337–359 (2013).
2. Willett, F. R., Avansino, D. T., Hochberg, L. R., Henderson, J. M. & Shenoy, K. V. High-performance brain-to-text communication via handwriting. *Nature* **593**, 249–254 (2021).
3. Geddes, C. E., Li, H. & Jin, X. Optogenetic editing reveals the hierarchical organization of learned action sequences. *Cell* **174**, 32–43 (2018).
4. Gallistel, C. R. *The Organization of Action: A New Synthesis* (Psychology Press, 2013).
5. Lashley, K. S. et al. *The Problem of Serial Order in Behavior* Vol. 21 (Bobbs-Merrill Oxford, 1951).
6. Khanna, A. R. et al. Single-neuronal elements of speech production in humans. *Nature* https://doi.org/10.1038/s41586-023-06982-w (2024).
7. Georgopoulos, A. P., Schwartz, A. B. & Kettner, R. E. Neuronal population coding of movement direction. *Science* **233**, 1416–1419 (1986).
8. Hochberg, L. R. et al. Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. *Nature* **485**, 372–375 (2012).
9. Collinger, J. L. et al. High-performance neuroprosthetic control by an individual with tetraplegia. *Lancet* **381**, 557–564 (2013).
10. Pandarinath, C. et al. High-performance communication by people with paralysis using an intracortical brain-computer interface. *eLife* **6**, e18554 (2017).
11. Inoue, Y., Mao, H., Suway, S. B., Orellana, J. & Schwartz, A. B. Decoding arm speed during reaching. *Nat. Commun.* **9**, 5243 (2018).
12. Hatsopoulos, N. G., Xu, Q. & Amit, Y. Encoding of movement fragments in the motor cortex. *J. Neurosci.* **27**, 5105–5114 (2007).
13. Suway, S. et al. Temporally segmented directionality in the motor cortex. *Cereb. Cortex* **28**, 2326–2339 (2018).
14. Qi, Y. et al. Dynamic ensemble Bayesian filter for robust control of a human brain-machine interface. *IEEE Trans. Biomed. Eng.* **69**, 3825–3835 (2022).
15. Gilja, V. et al. A high-performance neural prosthesis enabled by control algorithm design. *Nat. Neurosci.* **15**, 1752–1757 (2012).
16. Truccolo, W., Friehs, G. M., Donoghue, J. P. & Hochberg, L. R. Primary motor cortex tuning to intended movement kinematics in humans with tetraplegia. *J. Neurosci.* **28**, 1163–1178 (2008).
17. Moran, D. W. & Schwartz, A. B. Motor cortical representation of speed and direction during reaching. *J. Neurophysiol.* **82**, 2676–2692 (1999).
18. Koay, S. A., Charles, A. S., Thiberge, S. Y., Brody, C. D. & Tank, D. W. Sequential and efficient neural-population coding of complex task information. *Neuron* **110**, 328–349 (2022).
19. Xie, Y. et al. Geometry of sequence working memory in macaque prefrontal cortex. *Science* **375**, 632–639 (2022).
20. Roemschied, F. A. et al. Flexible circuit mechanisms for context-dependent song sequencing. *Nature* **622**, 794–801 (2023).
21. Ding, N., Melloni, L., Zhang, H., Tian, X. & Poeppel, D. Cortical tracking of hierarchical linguistic structures in connected speech. *Nat. Neurosci.* **19**, 158–164 (2016).

22. Poeppel, D., & Embick, D. Defining the relation between lnguistics and neuroscience. In *Twenty-first Century Psycholinguistics: Four Cornerstones* (ed Cutler, A.) 103–118 (Routledge, 2005).
23. Kadmon Harpaz, N., Ungarish, D., Hatsopoulos, N. G. & Flash, T. Movement decomposition in the primary motor cortex. *Cereb. Cortex* **29**, 1619–1633 (2019).
24. Georgopoulos, A. P., Ashe, J., Smyrnis, N. & Taira, M. The motor cortex and the coding of force. *Science* **256**, 1692–1695 (1992).
25. Kakei, S., Hoffman, D. S. & Strick, P. L. Muscle and movement representations in the primary motor cortex. *Science* **285**, 2136–2139 (1999).
26. Griffin, D. M., Hoffman, D. S. & Strick, P. L. Corticomotoneuronal cells are 'functionally tuned'. *Science* **350**, 667–670 (2015).
27. Schwartz, A. B. & Moran, D. W. Arm trajectory and representation of movement processing in motor cortical activity. *Eur. J. Neurosci.* **12**, 1851–1856 (2000).
28. Aflalo, T. N. & Graziano, M. S. Partial tuning of motor cortex neurons to final posture in a free-moving paradigm. *Proc. Natl Acad. Sci. USA* **103**, 2909–2914 (2006).
29. Omlor, W. et al. Context-dependent limb movement encoding in neuronal populations of motor cortex. *Nat. Commun.* **10**, 4812 (2019).
30. Terada, S.-I., Kobayashi, K. & Matsuzaki, M. Transition of distinct context-dependent ensembles from secondary to primary motor cortex in skilled motor performance. *Cell Rep.* **41**, 111494 (2022).
31. Xing, D., Truccolo, W. & Borton, D. A. Emergence of distinct neural subspaces in motor cortical dynamics during volitional adjustments of ongoing locomotion. *J. Neurosci.* **42**, 9142–9157 (2022).
32. Rouse, A. G. & Schieber, M. H. Condition-dependent neural dimensions progressively shift during reach to grasp. *Cell Rep.* **25**, 3158–3168 (2018).
33. Suresh, A. K. et al. Neural population dynamics in motor cortex are different for reach and grasp. *eLife* **9**, e58848 (2020).
34. Krebs, H. I., Aisen, M. L., Volpe, B. T. & Hogan, N. Quantization of continuous arm movements in humans with brain injury. *Proc. Natl Acad. Sci. USA* **96**, 4645–4649 (1999).
35. Sergio, L. E., Hamel-Pâquet, C. & Kalaska, J. F. Motor cortex neural correlates of output kinematics and kinetics during isometric-force and arm-reaching tasks. *J. Neurophysiol.* **94**, 2353–2378 (2005).
36. Pandarinath, C. et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nat. Methods* **15**, 805–815 (2018).
37. Shenoy, K. V. & Kao, J. C. Measurement, manipulation and modeling of brain-wide neural population dynamics. *Nat. Commun.* **12**, 633 (2021).
38. Vargas-Irwin, C. E. et al. Watch, imagine, attempt: motor cortex single-unit activity reveals context-dependent movement encoding in humans with tetraplegia. *Front. Hum. Neurosci.* **12**, 450 (2018).
39. Dushanova, J. & Donoghue, J. Neurons in primary motor cortex engaged during action observation. *Eur. J. Neurosci.* **31**, 386–398 (2010).
40. Soechting, J. F. & Lacquaniti, F. Invariant characteristics of a pointing movement in man. *J. Neurosci.* **1**, 710–720 (1981).
41. Viviani, P. & Terzuolo, C. Trajectory determines movement dynamics. *Neuroscience* **7**, 431–437 (1982).
42. Lacquaniti, F., Terzuolo, C. & Viviani, P. The law relating the kinematic and figural aspects of drawing movements. *Acta Psychol.* **54**, 115–130 (1983).
43. Edelman, S. & Flash, T. A model of handwriting. *Biol. Cybern.* **57**, 25–36 (1987).
44. Mussa-Ivaldi, F. A., Giszter, S. F. & Bizzi, E. Linear combinations of primitives in vertebrate motor control. *Proc. Natl Acad. Sci. USA* **91**, 7534–7538 (1994).
45. d'Avella, A., Saltiel, P. & Bizzi, E. Combinations of muscle synergies in the construction of a natural motor behavior. *Nat. Neurosci.* **6**, 300–308 (2003).
46. Okorokova, E., Lebedev, M., Linderman, M. & Ossadtchi, A. A dynamical model improves reconstruction of handwriting from multichannel electromyographic recordings. *Front. Neurosci.* **9**, 389 (2015).
47. Lebedev, M. A. et al. in *Brain–Computer Interface Research: A State-of-the-Art Summary 8* (eds Guger, C. et al.) 11–23 (Springer, 2020).
48. Linderman, M., Lebedev, M. A. & Erlichman, J. S. Recognition of handwriting from electromyography. *PLoS ONE* **4**, e6791 (2009).
49. Jiang, H. et al. Short report: surgery for implantable brain-computer interface assisted by robotic navigation system. *Acta Neurochir.* **164**, 2299–2302 (2022).
50. Qi, Y., Liu, B., Wang, Y. & Pan, G. Dynamic ensemble modeling approach to nonstationary neural decoding in brain-computer interfaces. *Adv. Neural Inf. Process. Syst.* **547**, 6089–6098 (2019).
51. Qi, Y., Zhu, X. & Xiong, X. Human motor cortex encodes complex handwriting through a sequence of stable neural states: source code. *Zenodo* https://zenodo.org/records/14865736 (2025).

Acknowledgements

We thank Xiang Li, Xufei Li, J. Yan, H. Wu, J. Qiao and Y. Hao for the technical support on the BCI system. This work was supported by grants from the National Key R&D Program of China (number 2018YFA0701400 (Y.W.)), the Key R&D Program of Zhejiang (number 2022C03011 (Y.W.) and 2023C03001 (Y.Q.)), the National Natural Science Foundation of China (number 62336007 (Y.W.) and number 62276228 (Y.Q.)), the Zhejiang Provincial Natural Science Foundation of China (number LR24F020002 (Y.Q.)) and the Starry Night Science Fund of Zhejiang University Shanghai Institute for Advanced Study (number SN-ZJU-SIAS-002 (Y.W.)). The funders had no role in study design, data collection and analysis, decision to publish or preparation of the paper.

Author contributions

Y.Q. conceived the project, proposed the state-dependent encoding model and TFC algorithm, designed the experiment, collected and analysed the data, and wrote the paper. X.Z. contributed to data analysis, conducted simulations and wrote the paper. X.X. contributed to the experimental system, collected and analysed the data, and wrote the paper. X.Y. contributed to data collection. N.D. and H.W. contributed to the paper writing. K.X., J. Zhu and J. Zhang contributed to the experiment design and implementation. Y.W. supervised the project and contributed to the paper writing.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41562-025-02157-x.

Correspondence and requests for materials should be addressed to Yu Qi, Junming Zhu or Yueming Wang.

Peer review information *Nature Human Behaviour* thanks Elizaveta Okorokova, Marc Schieber and the other, anonymous, reviewer(s) for their contribution to the peer review of this work. Peer reviewer reports are available.

nature portfolio

Corresponding author(s): Yu Qi, Junming Zhu, Yueming Wang

Last updated by author(s): Feb 13, 2025

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☐	☒	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☐	☒	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☐	☒	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☐	☒	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☐	☒	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	Data were collected using the Central Suite software package v7.0.4 with the Neuroport System (Blackrock Microsystems). The graphical user interface (GUI) of the experimental task was developed in Python 3.9.
Data analysis	Statistical analysis were performed using MATLAB R2022a, Python 3.9, and Graphpad Prism 9. Spike sorting were performed with either Plexon Offline Spike Sorter v4 (offline analysis) or Central Suite software package v7.0.4 (online decoding).

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Portfolio guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A description of any restrictions on data availability
- For clinical datasets or third party data, please ensure that the statement adheres to our policy

Data relevant to this study are accessible under restricted access according to our clinical trial protocol. Access can be granted upon request to Dr. Yu Qi

(qiyu@zju.edu.cn). The response can be expected within three weeks. Any data provided must be kept confidential and must not be shared with others without approval. Source data to reproduce the figures in the manuscript are provided in this paper.

Human research participants

Policy information about studies involving human research participants and Sex and Gender in Research.

Reporting on sex and gender	The single participant is a right-handed male, 75-year-old at the time of data collection.
Population characteristics	This study included one participant, who is a right-handed man, 75-year-old at the time of data collection.
Recruitment	Participants were recruited through the Second Affiliated Hospital of Zhejiang University. Enrollment was entirely dependent on the specific criteria of the clinical protocol. Thus, we do not expect any significant self-selection bias in this study.
Ethics oversight	All clinical and experimental procedures conducted in this study received approval from the Medical Ethics Committee of The Second Affiliated Hospital of Zhejiang University (Ethical review number 2019-158, approved on 05/22/2019) and were registered in the Chinese Clinical Trial Registry (Ref. ChiCTR2100050705). This study used an observational design and no intervention took place that was not driven by clinical need. Informed consent was obtained verbally from the participant, along with the consent of his family members, and was duly signed by his legal representative.

Note that full information on the approval of the study protocol must also be provided in the manuscript.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	The sample sizes in this study were determined by the amount of available data with the participant, and the estimation of how much data would be required to perform statistical comparisons.
Data exclusions	Data were not excluded from the analysis.
Replication	The main experiments were carried out between June 2022 and July 2023, mostly two or three days per week. Most experimental days contained two sessions with the same characters, and the experiment sessions on different days contained different characters.
Randomization	Not relevant. Only one participant was included in the study.
Blinding	Not relevant. Only one participant was included in the study.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☒	☐ Animals and other organisms
☐	☒ Clinical data
☒	☐ Dual use research of concern

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Clinical data

Policy information about clinical studies

All manuscripts should comply with the ICMJE guidelines for publication of clinical research and a completed CONSORT checklist must be included with all submissions.

Clinical trial registration	Ref. ChiCTR2100050705
Study protocol	A description of the study protocol can be found on the ChiCTR website at https://www.chictr.org.cn/showproj.html?proj=130829.
Data collection	Data collection was carried out in a dedicated ward of the Second Affiliated Hospital of Zhejiang University. The participant was recruited in August 2019. Data used in this study were collected between June 2022 and July 2023.
Outcomes	This clinical trial was to evaluate the potential of brain-computer interfaces for motor reconstruction and restoration. Since the clinical trial is a phase 0 exploratory clinical trial, the secondary outcomes were not predefined. The predefined primary endpoint was to enable brain-computer interface control, which would be measured to assess the efficacy of the brain-computer interface during motor control.

nature human behaviour

Supplementary information

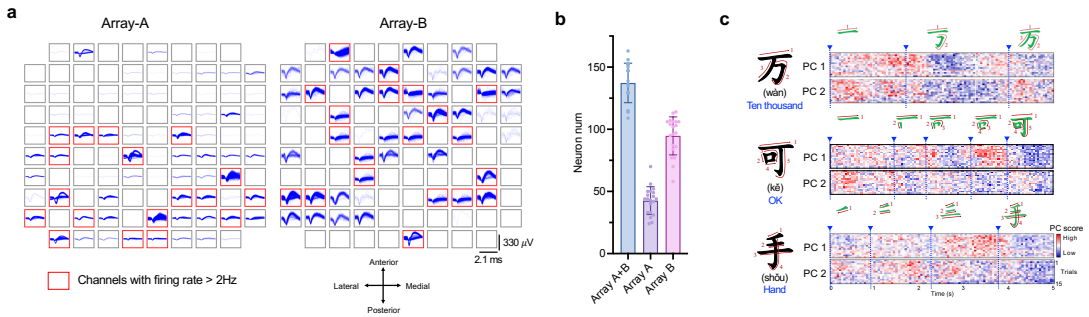
https://doi.org/10.1038/s41562-025-02157-x

Human motor cortex encodes complex handwriting through a sequence of stable neural states

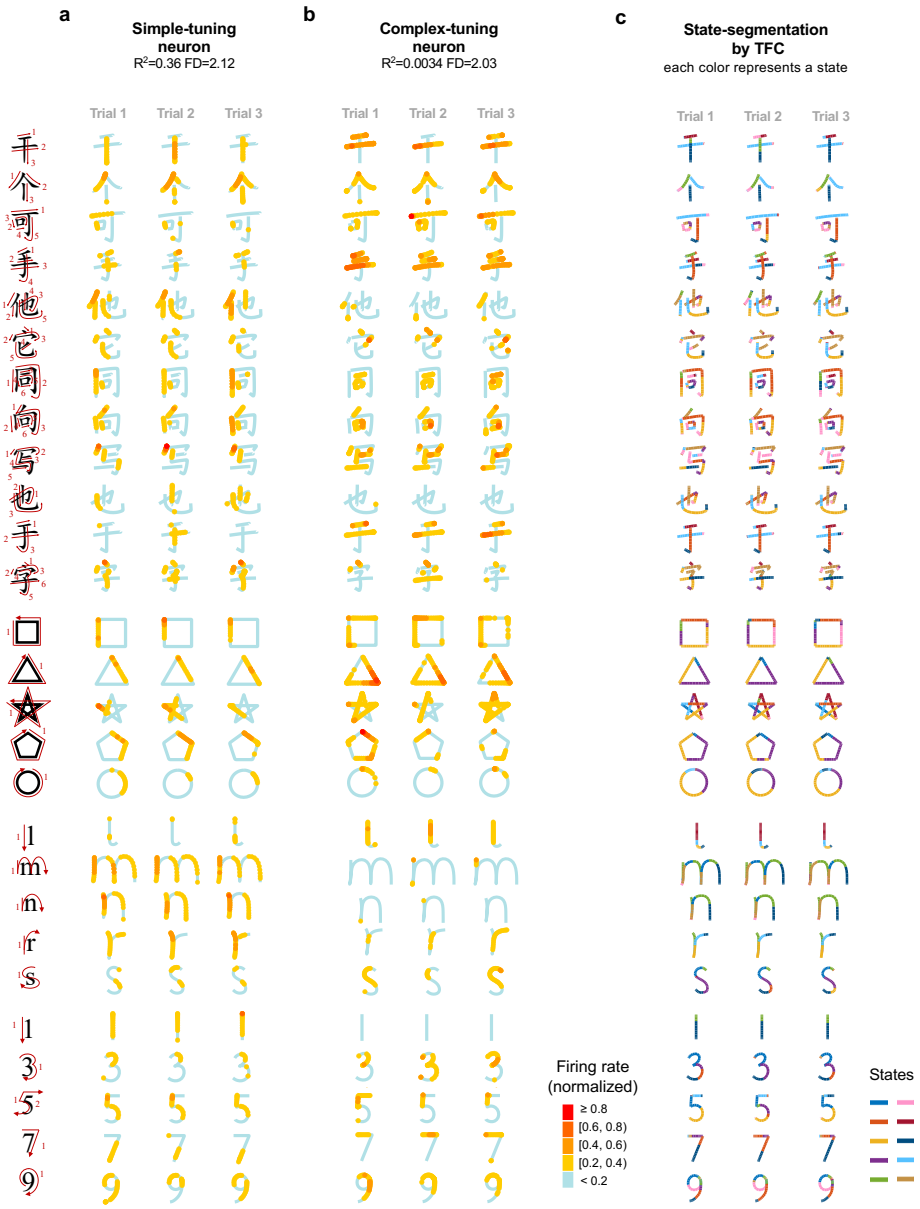
In the format provided by the authors and unedited

Supplementary Information

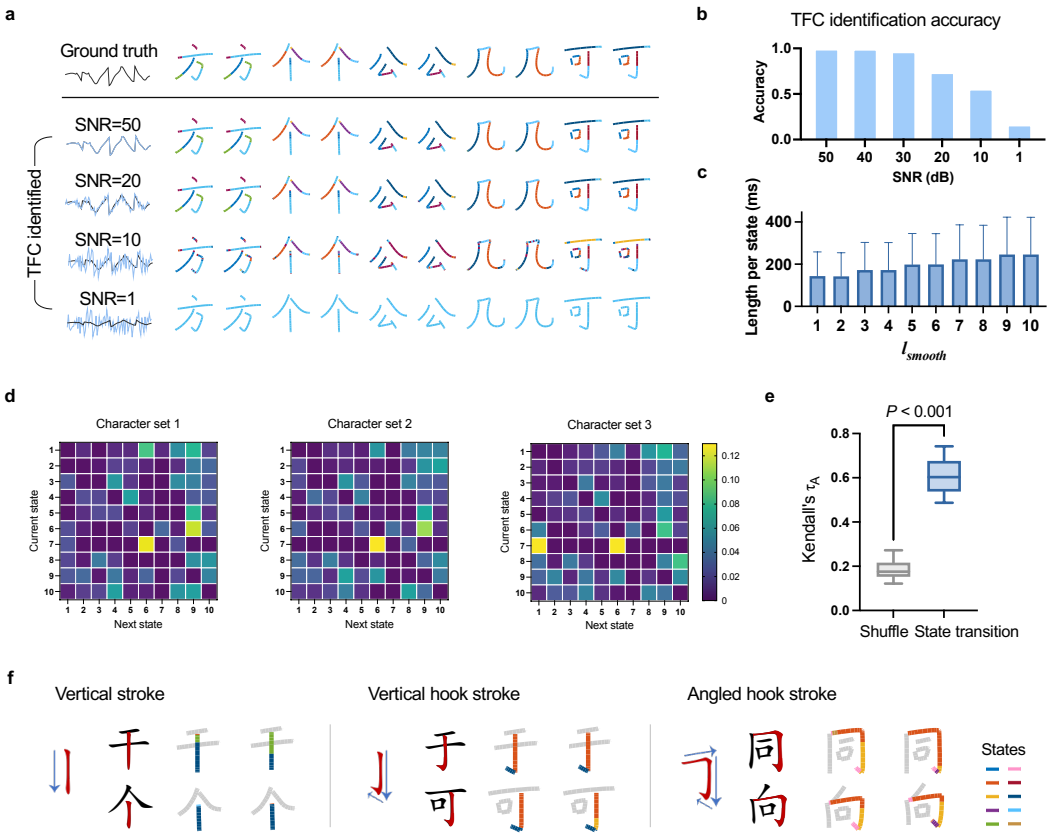
Supplementary Figures



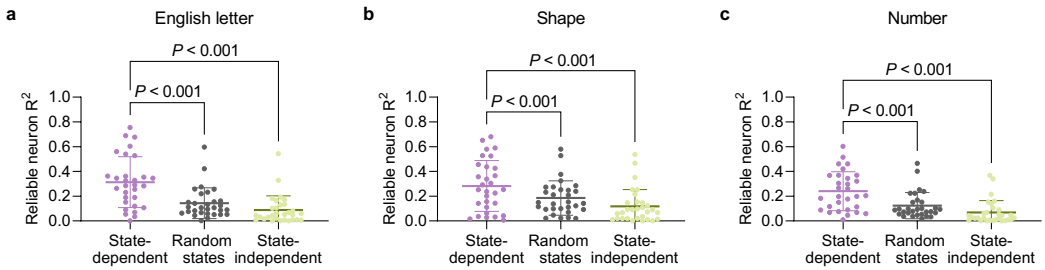
Supplementary Fig. 1 | Example of neural signal recordings and the neural representation of handwriting. a, Illustration of spiking activity recorded from each microelectrode array during a 3-minute window, captured on the 1402nd day after array implantation. **b**, Statistics of single unit numbers for both arrays across 20 sessions (M±SD). There were 137.3±15.8 units after offline spike sorting with Array A and B together (with 42.6±11.2 for Array A and 94.7±15.2 for Array B, n = 20 sessions). **c**, Neural activity in the top 2 principal components is shown for three example characters, 15 repetitions for each character. The color scale is normalized within each panel separately for visualization.



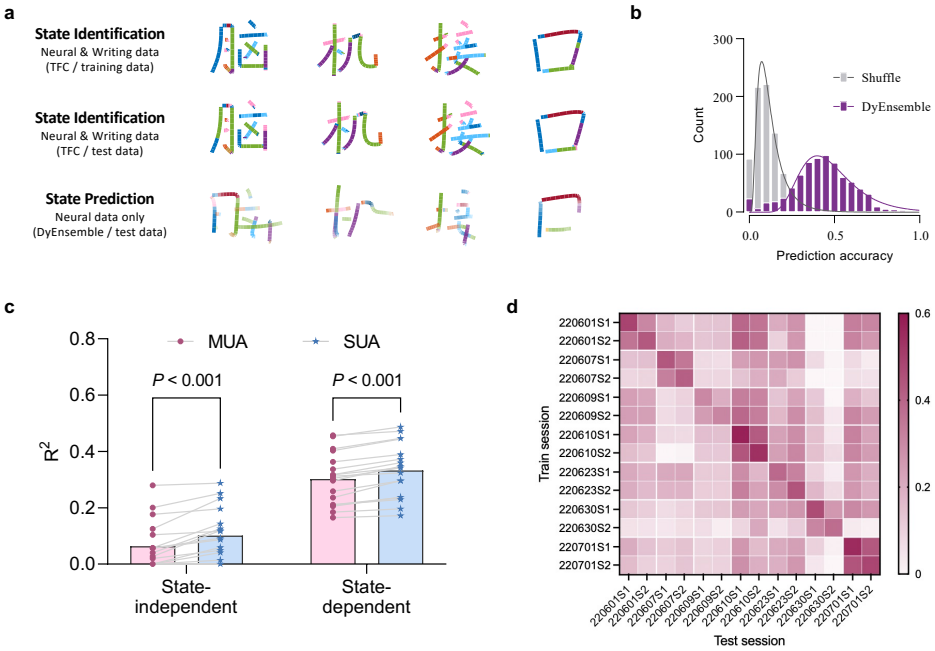
Supplementary Fig. 2 | Examples of neuronal responses and state segmentation during handwriting of different contents. a, Illustrative examples showcasing the neural response of a simple-tuning neuron. The neuron demonstrates a consistent preferred direction of down left, while it only selectively fires at part of down left trajectories. This neural activity can be explained by a state-dependent model, where each state shares a consistent preferred direction with diverse modulation depth. **b**, Similar to **a**, but with a complex-tuning neuron, this neuron exhibits divergent preferred directions and fires in both leftward, rightward, and downward trajectories. This neural activity can be explained by a state-dependent directional tuning model where the preferred direction shifts across states. **c**, State-segmentation results with the TFC method across different characters in three repetitions.



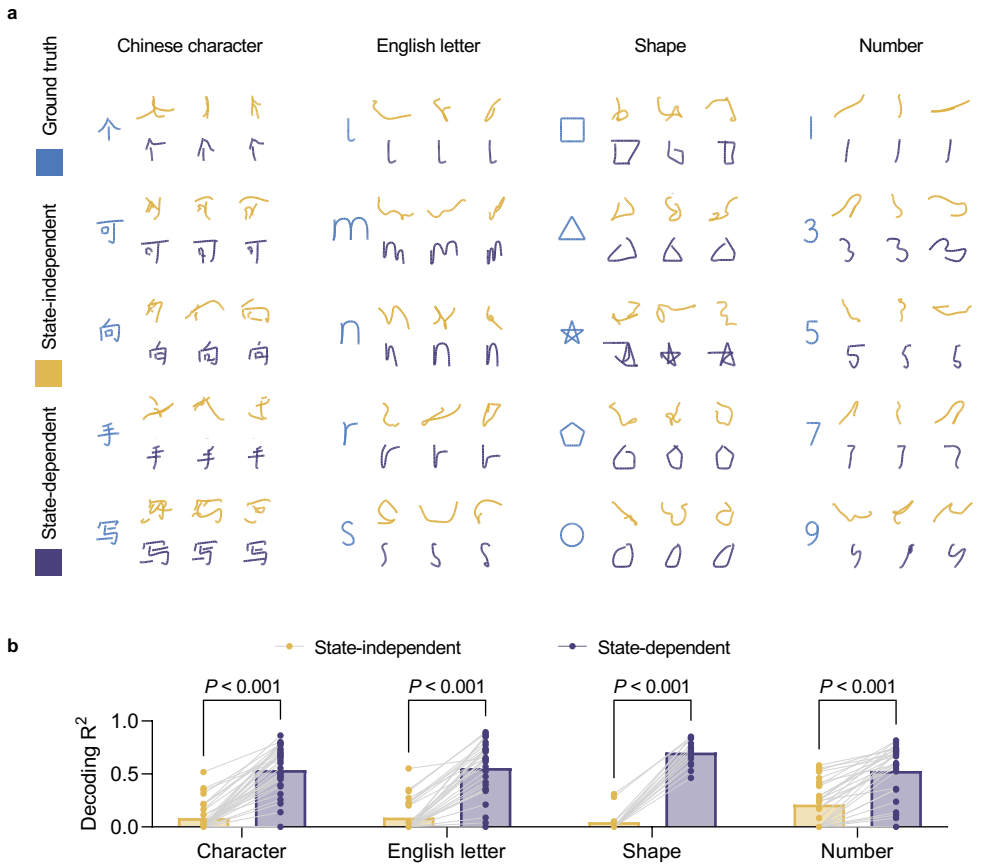
Supplementary Fig. 3 | Evaluation of the TFC algorithm in writing state identification. a, Simulation experiment assessing the TFC algorithm's efficacy in correctly identifying distinct mode (state) and mode (state) switch processes during handwriting (see details in Supplementary Text). **b**, State identification accuracy under varying signal-to-noise ratios (SNR). **c**, Statistics of the length of states with different selections of l_{smooth}. **d**, The state-transition matrices with three non-overlapping character sets (session 220623-S5). Each element (i, j) indicates the state transfer probability from state i to state j. The diagonal values are set to 0. **e**, The similarity of state-transition matrices of each pair of non-overlapping character sets within sessions (n = 18), measured by the Kendall's τ_A value. The Kendall's τ_A values are significantly higher than the shuffle condition (paired two-tailed t-test, $t(17) = 18.21$, $P = 1.373\text{E-}12$, Cohen's d = 6.82), indicating state-transition patterns independent of character sets. **f**, The state sequences during writing three exemplar strokes. We only focused on the highlighted strokes (colored), and the other parts of the characters were left gray. Consistent state sequences were observed when writing the same stroke, regardless of the Chinese character in which the stroke appeared.



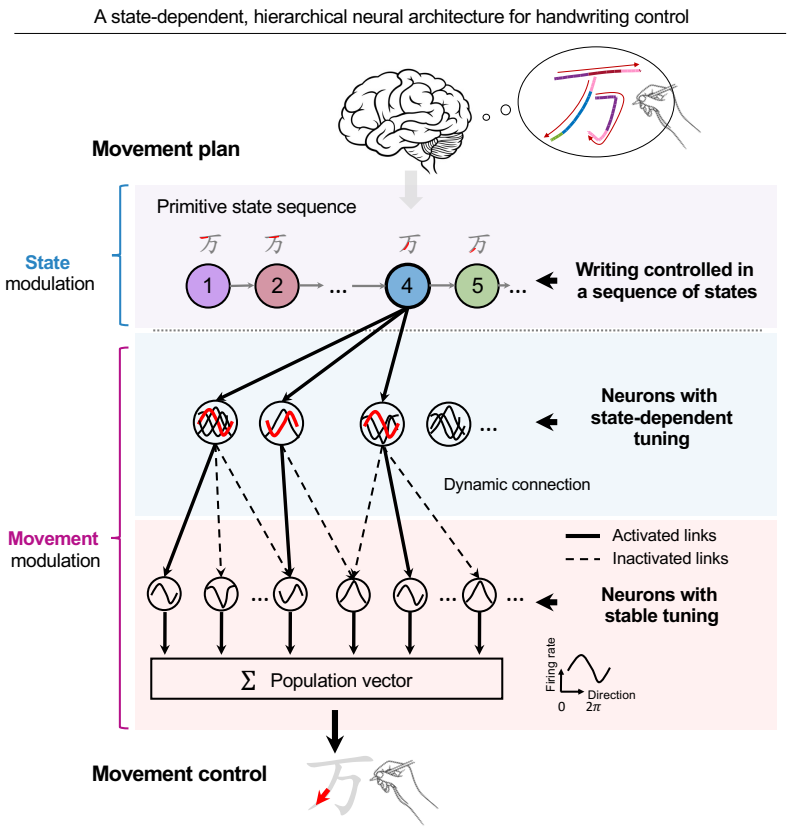
Supplementary Fig. 4 | The state-dependent encoding model better predicts neuronal activities when writing different content. a-c, The state-dependent directional tuning model significantly enhances the R^2 for reliably tuned neurons (n = 30) with the writing of English letters (**a**, State-dependent vs. Random states: paired two-tailed *t*-test, $t(29) = 6.747$, $P = 2.098\text{E-}7$, Cohen's d = 0.997; State-dependent vs. State-independent: paired two-tailed *t*-test, $t(29) = 7.718$, $P = 1.642\text{E-}8$, Cohen's d = 1.348), shapes (**b**, State-dependent vs. Random states: paired two-tailed *t*-test, $t(29) = 6.747$, $P = 2.098\text{E-}7$, Cohen's d = 0.997; State-dependent vs. State-independent: paired two-tailed *t*-test, $t(29) = 7.718$, $P = 1.642\text{E-}8$, Cohen's d = 1.348), and numbers (**c,** State-dependent vs. Random states: paired two-tailed *t*-test, $t(29) = 6.747$, $P = 2.098\text{E-}7$, Cohen's d = 0.997; State-dependent vs. State-independent: paired two-tailed *t*-test, $t(29) = 7.718$, $P = 1.642\text{E-}8$, Cohen's d = 1.348).



Supplementary Fig. 5 | State prediction with the state-dependent decoding model. a, State identification with TFC, with the training set (first line), test set (second line, without model learning, directly applied the learned models during training). The third line illustrates the state predictions in a state-dependent decoding process with the neural data only. **b**, The state prediction accuracy of the state-dependent neural decoder is shown in the histogram (n = 800, 22 sessions with leave-one-repeat-out cross-validation; please see Supplementary Text for details). **c**, Comparison of handwriting decoding performance with single-unit activity (SUA, offline sorted) and multi-unit activity (MUA, unsorted). The mean difference of R^2 was minor (0.33 vs. 0.30 for state-dependent and 0.10 vs. 0.06 for state-independent, n = 18 sessions. State-independent: paired two-tailed *t*-test, $t(17) = 4.453$, $P = 3.49E\text{-}4$, Cohen's d = 0.426; State-dependent: paired two-tailed *t*-test, $t(17) = 9.054$, $P = 6.507E\text{-}8$, Cohen's d = 0.330). **d**, The cross-session/day decoding performance with the state-dependent decoder, with a month time span.



Supplementary Fig. 6 | Comparison of state-dependent and state-independent handwriting decoding with different writing contents. a, Comparison of the decoded handwriting trajectories. **b**, Comparison of the decoding performance with the R^2 (Character: paired two-tailed *t*-test, n = 36, $t(35) = 12.23$, $P = 3.385E\text{-}14$, Cohen's d = 2.232; English letter: paired two-tailed *t*-test, n = 30, $t(29) = 11.38$, $P = 3.229E\text{-}12$, Cohen's d = 2.271; Shape: paired two-tailed *t*-test, n = 15, $t(14) = 23.05$, $P = 1.567E\text{-}12$, Cohen's d = 6.341; Number: paired two-tailed *t*-test, n = 30, $t(29) = 9.012$, $P = 6.614E\text{-}10$, Cohen's d = 1.396).



Supplementary Fig. 7 | Neural processing architecture of handwriting. Sophisticated handwriting is divided into stable states, each corresponding to a stroke fragment. Under each state, movement is controlled through a state-specific neural ensemble, which possibly consists of two types of neurons. The complex tuning neurons encode a fragment of the movement trajectory and are connected to a set of simple tuning neurons, which directly control the movement direction at the current time moment.

Supplementary Text

1 Data collection sessions

We scheduled 2-3 experiment days in a week, and each experimental day contained 2-3 sessions. During a session, the participant sat in a wheelchair in an upright position, with a pillow placed to support his head and neck. His hand rested on a dinner board in front of him. A computer monitor was placed about 1 meter in front of him, indicating which character to write. The experimental sessions used for analysis in this work are listed in Supplementary Table 1. The sessions used in each figure are listed in Supplementary Table 2.

Supplementary Table 1. Data sessions included in this study.

Session	N_C	N_R	Characters	P
220601-S1	30	3	成吃此的定对多法还好和会看来里没那你年如时是我行学永有者这作	
220601-S2	30	3	成吃此的定对多法还好和会看来里没那你年如时是我行学永有者这作	
220607-S1	30	3	把本但当地点动而发分过后话回活间见经开老们身世所同位用知总走	
220607-S2	30	3	把本但当地点动而发分过后话回活间见经开老们身世所同位用知总走	
220609-S1	30	3	爱被常到道得第都高给国果孩家理美面其起前亲使说她西现些样因种	
220609-S2	30	3	爱被常到道得第都高给国果孩家理美面其起前亲使说她西现些样因种	
220610-S1	30	3	不出次从大儿尔方个公几可名女去什生手他天头外为问无也在长中自	
220610-S2	30	3	不出次从大儿尔方个公几可名女去什生手他天头外为问无也在长中自	
220623-S4	30	3	报并场处传风告各更光合何画欢机近决军空拉利论罗妈男品求全任实	CW
220623-S5	30	3	报并场处传风告各更光合何画欢机近决军空拉利论罗妈男品求全任实	
220630-S1	30	3	变表别读房放非该花或结界金觉科苦快连林流命母呢轻师思体条往物	
220630-S2	30	3	变表别读房放非该花或结界金觉科苦快连林流命母呢轻师思体条往物	
220701-S1	30	3	安必边布车达代电东夫关化即加交克吗民目内平却让色声失始受书司	
220701-S2	30	3	安必边布车达代电东夫关化即加交克吗民目内平却让色声失始受书司	
220707-S1	30	3	巴白比产打反父火及记今乐立马片气切认水死四岁太听万王五向写由	
220707-S2	30	3	巴白比产打反父火及记今乐立马片气切认水死四岁太听万王五向写由	
220708-S1	30	3	完先笑信星性许言业医音应友语元员远约月运再早则怎战找至主字坐	
220708-S2	30	3	完先笑信星性许言业医音应友语元员远约月运再早则怎战找至主字坐	
221027-S1	6	15	川万可什公手	CW
230327-S2	8	5	浙江大学脑机接口	SW
230511-S3	32	3	埋坡坏坤环理玻珅权杈杯枓杆种和私科秆汉叹仅扠钗汊呔怯汰钛伏吠沃状	CW
230724-S3	37	3	干个可手他它同向写也于字, 5 shapes, 10 numbers, 10 English letters (k-t)	CW

* N_C: The number of characters
* N_R: The number of repeats

* P: Paradigm

* CW: Single-character writing with visual guidance paradigm

* SW: Sentence writing with visual guidance paradigm

Supplementary Table 2. List of all data collection sessions used for the figures

Figures	Sessions
Fig. 2a, 3f, 5a, 5b, 5c, 5d, 5e, S1b, S3c	220601-S1; 220601-S2; 220607-S1; 220607-S2; 220609-S1; 220609-S2; 220610-S1; 220610-S2; 220623-S4; 220623-S5; 220630-S1; 220630-S2; 220701-S1; 220701-S2; 220707-S1; 220707-S2; 220708-S1; 220708-S2; 230511-S3; 230724-S3
Fig. 2b, 2c, 2d, S2a, S2b, S2c, S3f, S4a, S4b, S4c	230724-S3
Fig. 3b, 3c, 3d, 3e, 3h, 4a, 4b, 4c, 6c, 6d, S3e, S5c, S5d	220601-S1; 220601-S2; 220607-S1; 220607-S2; 220609-S1; 220609-S2; 220610-S1; 220610-S2; 220623-S4; 220623-S5; 220630-S1; 220630-S2; 220701-S1; 220701-S2; 220707-S1; 220707-S2; 220708-S1; 220708-S2
Fig. 3g	220601-S1
Fig. 6b, S5a, S5b, S6a, S6b	230327-S2
Fig. S1c	221027-S1
Fig. S3a, S3b	220610-S1
Fig. S3d	220623-S4

2 Neural representation of handwriting

2.1 The tuning behavior of neurons (Fig. 2a)

To evaluate the tuning behavior of neurons, we assessed it using two metrics: R^2 and FD. A total of 20 experimental sessions were included (please refer to Supplementary Tables 1 and 2), involving 2850 neurons. Note that, since we plotted neurons from different sessions together, it may contain reduplicated neurons appeared at multiple sessions.

The R^2 of one neuron represents the proportion of its neural variance explained by the directional tuning model. The spiking probability P(spike|velocity) is modeled as a linear function of the velocity components along the x and y axes, given by:

$$P(spike|velocity) = b_0 + b_x v_x + b_y v_y \quad (1)$$

where v_x and v_y are the velocity along x- and y-axes, respectively, and b_0, b_x, b_y are the

model parameters. The R^2 was computed by:

$$R^2 = 1 - \frac{\Sigma_k (y_k - \hat{y}_k)^2}{\Sigma_k (y_k - \bar{y})^2} \tag{2}$$

where y_k is the firing rate at time step k of the analyzed neuron, $\hat{y}_k$ is the firing rate estimated by the directional tuning model at time step k, and $\bar{y}$ is the average firing rate of this neuron.

The FD (Fisher's discriminant value) is a metric indicating one neuron's discriminative ability (or reliability) of the tuning patterns by comparing the inter-character distance to the intra-character distance of neural representations, which was defined as:

$$FD_c = \frac{d_{inter}^c}{d_{intra}^c} = \frac{\Sigma_{C_l=c, C_m \neq c} (y_l - y_m)^2 / N_{(l,m)}}{\Sigma_{C_l=c, C_n=c} (y_l - y_n)^2 / N_{(l,n)}} \tag{3}$$

$$FD = \frac{1}{N_C} \Sigma_C FD_C \tag{4}$$

where c denotes a certain Chinese character and N_C denotes the total number of the target characters. The term d_{inter}^c represents the average neural distance between different characters , while d_{intra}^c demotes the average neural distance between different repetitions of the same character. l, m, n are trial indexes and $\Sigma_{C_l=c, C_m \neq c} (y_l - y_m)^2$ refers to the sum over all (l, m) pairs, which satisfy trial l's target C_l is character c while trial m's target C_m is not character c. $N_{(l,m)}$ represents the total number of (l, m) pairs that satisfy this condition. A high FD value indicates a high discriminative ability of neuronal response, namely consistent in repetitions while discriminative with different characters.

2.2 The rasters overlaid on characters (Fig. 2b, 2c, 2d, S2a, S2b)

To plot a raster overlaid on a character, spike counts of the example neurons were first binned into 50 ms bins. Then, max-min normalization was applied to the neuronal firing rate in bins. Next, the sequence of neural responses was plotted on the corresponding trajectory of the character. Here, the handwriting trajectory is the same as the visual instruction in the video. Only strong neural responses above a certain threshold (0.2) were displayed for clearer visualization.

3 Identification of stable states

3.1 Simulation for TFC evaluation (Fig. S3a, S3b)

To validate the reliability of the TFC algorithm, we first conducted experiments on simulated data, as shown in Fig. S3a and S3b. The simulated experiments utilized the character set from 220610-S1 consisting of 30 Chinese characters. Initially, each character was randomly divided into multiple segments, with lengths varying between 5 and 20 bins. Subsequently, we randomly generated $N_{\text{state}} = 10$ encoding models, supposing there are 10 states in the writing process. Each segment was then randomly assigned to one encoding model, and the corresponding neural signals were generated using this model and the kinematic segment. Finally, the neural signals for each character were constructed by concatenating the neural signals generated by different encoding models, with

the addition of Gaussian noise at different levels. Note that we generated three sets of neural signals for each character, with the same model assignments but random noise to mimic the real experimental settings.

Specifically, for each encoding model, we first generated cosine tuning curves for $d_y = 100$ simulated neurons by drawing their preferred directions (PDs) randomly from a uniform distribution over $[0, 2\pi]$. Baseline firing rates were drawn from a uniform distribution over [1, 2.5], and modulation depths were drawn from a uniform distribution over [0.005, 0.025], where these parameter ranges were determined based on observations of real data. Then the encoding model can be constructed as a linear mapping matrix $B \in \mathbb{R}^{N\times(1+d_x)}$, where $d_x = 2$ is the dimension of kinematics (writing velocities along x- and y-axes), 1 denotes the dimension of bias. Here, the first column of the mapping matrix B is the baseline firing rate of each neuron, while the second and third columns of B are the sine and cosine values of neuron PDs, multiplied by the corresponding modulation depth. The above process was repeated 10 times to obtain 10 encoding models. These encoding models allow us to predict the firing rates based on the kinematic inputs.

Different levels of Gaussian white noise were added to the final constructed neural signals, according to the signal-to-noise ratio (SNR):

$$\mathrm{SNR_{dB}} = 10\log_{10}\left(\frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}}\right) = 10\log_{10}\frac{\Sigma_k y_k^2}{\Sigma_k(\tilde{y}_k - y_k)^2} \tag{5}$$

where y_k is the clean neural signals at time step k, and $\tilde{y}_k$ is the neural signals with added noise, which can be obtained directly from the function awgn(.) in MATLAB. In our simulation, we set up 5 noise levels of $\mathrm{SNR}_{dB} = 10, 20, 30, 40, and\ 50$, respectively.

After generating the simulation data, we then used the TFC algorithm to identify these 10 states using $M = 10$ models. The error smooth window size l_{smooth} was 1. In Fig. S3b, the accuracy of model assignment was reported according to:

$$acc_{\mathcal{H}_m} = \frac{\max\left(T_{\mathcal{H}_m,1}, T_{\mathcal{H}_m,2}, \dots, T_{\mathcal{H}_m,j}, \dots, T_{\mathcal{H}_m,N_{\mathrm{state}}}\right)}{T_{\mathcal{H}_m}} \tag{6}$$

$$acc = \frac{\sum_{m=1}^{M} acc_{\mathcal{H}_m}}{M} \tag{7}$$

where $T_{\mathcal{H}_m,j}$ represents the number of overlapping indices between the data selected by model $\mathcal{H}_m$ and the data of state j. $T_{\mathcal{H}_m}$ denotes the length of data selected by model $\mathcal{H}_m$.

3.2 Identifying stable states with TFC (Fig. S2c, S3f)

We used the TFC algorithm to analyze real data and segmented the writing process into stable states. Fig. S2c, S3f shows the examples from 230724-S3. Before applying the TFC algorithm, the spike counts were divided into 200 ms bins using a sliding window with a step of 50 ms. A moving average of 10 bins was then applied for smoothing. The neural signals corresponding to the writing phase were extracted for analysis. The TFC algorithm was configured with $M = 10$ models. The error smooth window size l_{smooth} was set to 5 bins.

3.3 The state-transition patterns (Fig. S3d, S3e)

To assess whether state transitions exhibit temporal dependencies, i.e., whether one state comes after another randomly or some states are more likely to come after a certain state, we divided the data from each session into three non-overlapped subsets and calculated the state transition probabilities for each subset. For each state transition matrix, we removed the diagonal elements that reflect self-transition within a state since they usually contain outlying large values. In S3d, we visualized the state transition matrices for the three subsets (with different character sets each) from an example session.

Subsequently, we computed the pairwise Kendall's τ_A values between the three subsets for all 18 sessions, where τ_A ranges from [-1, 1], with higher values indicating greater similarity between the transition matrices. To obtain the shuffled control, Kendall's τ_A was recalculated after randomly shuffling the transition matrices.

4 State-dependent neural encoding of handwriting

4.1 Neural encoding loss with different model numbers in TFC (Fig. 3b, 3c, 3d, 3e)

To investigate the relationship between the neural encoding capability of the TFC algorithm and the number of specified models, we varied the number of models from 1 to 100 and plotted the resulting loss curve for neural encoding. The experiment included 18 sessions of data, as described in Supplementary Tables 1 and 2. Data were first divided into a training set and a test set, with three-fold cross-validation with no overlapping characters. That is, each session contains 30 characters with 3 repeats, and we divided them into 3 folds, and each fold contains 20 characters for training and the other 10 characters for test. The figure plots the average encoding loss for both the training set and test set with each model number (n = 18 sessions), with the error bar denoting the standard deviation value.

Besides, we also plotted the loss decrease for both the training set and test set, which is calculated from the difference of the loss value. To choose an appropriate number of models, we computed the BIC (Bayesian Information Criterion) value using the following equation:

$$\mathrm{BIC} = -2\ln(L) + k \cdot \ln(n) \tag{8}$$

where L is the likelihood of the model, and k is the number of parameters in the model, n is the number of data points. Since we already have the value of the encoding loss, i.e., the mean squared error (MSE), the only thing we need to do is to calculate the $ln(L)$. Assuming the residuals follow a Gaussian distribution with zero mean, then the variance can be estimated using the mean squared error, MSE. Thus, the log-likelihood can be calculated as follows:

$$\ln(L) = \ln \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i-\hat{y}_i)^2}{2\sigma^2}} \tag{9}$$

$$= \sum_{i=1}^{n} \ln \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i-\hat{y}_i)^2}{2\sigma^2}}$$

$$
\begin{aligned}
&= \sum_{i=1}^{n} -\ln\sqrt{2\pi\sigma^2} - \frac{(y_i - \hat{y}_i)^2}{2\sigma^2} \\
&= -\frac{n}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \\
&= -\frac{n}{2}\ln(2\pi\sigma^2) - \frac{n}{2} \\
&= -\frac{n}{2}\ln(2\pi \cdot \mathrm{MSE}) - \frac{n}{2}.
\end{aligned}
$$

Here, y_i is the ground truth of neural signals, and $\hat{y}_i$ is the neural signals estimated by our models. So the final BIC can be calculated directly from the encoding loss:

$$\mathrm{BIC} = n \cdot \ln(2\pi \cdot \mathrm{MSE}) + n + k \cdot \ln(n). \tag{10}$$

4.2 The pair-wise encoding loss across states (Fig. 3f)

In Fig. 3f, we calculated the encoding loss between each pair of states, namely the models in TFC. Our analysis was based on 20 experimental sessions, where each session involved the collection of at least 30 Chinese characters with 3 repetitions (please refer to Supplementary Tables 1 and 2). We employed the leave-one-repeat-out method, resulting in a total of 60 runs of the TFC algorithm. And the reported loss was the average across these runs.

Here, we calculate each pair of encoding loss $\mathcal{L}(i,j)$ between state i and state j. The computation of $\mathcal{L}(i,j)$ involved encoding the kinematic data selected by model $\mathcal{H}_i$ using model $\mathcal{H}_j$. Subsequently, we calculated the mean squared error between the encoded neural signals and the neural signals selected by model $\mathcal{H}_i$. Then the final loss was averaged over all time bins and neurons. This process can be mathematically expressed as:

$$\mathcal{L}(i,j) = \sum_{n=1}^{d_y}\sum_{k=1}^{T_{\mathcal{H}_i}} \sqrt{\left(Y_{\mathcal{H}_i} - \mathcal{H}_j\left(X_{\mathcal{H}_i}\right)\right)^2}. \tag{11}$$

Here, $Y_{\mathcal{H}_i} \in \mathbb{R}^{d_y \times T_{\mathcal{H}_i}}$ represents the neural signals selected by model $\mathcal{H}_i$, $\mathcal{H}_j(\cdot)$ denotes the encoding function of model $\mathcal{H}_j$, and $X_{\mathcal{H}_i}$ corresponds to the kinematic data selected by model $\mathcal{H}_i$.

4.3 Directional tuning curves in different states (Fig. 2b, 2c, 2d, 5b, 5c)

To closely examine the responses from reliably tuning neurons ($R^2 \geq 0.1$ and FD ≥ 1.4, n = 179) out of 20 experimental sessions, we plotted the tuning curves of these neurons with and without considering the states. Fig. 5b and 5c display four example neurons from session 230724-S3, including two simple-tuning neurons and two complex-tuning neurons.

Regarding the specifics of drawing the tuning curve, the movement kinematics during writing were meticulously partitioned into eight directional intervals (0°~45°, 45°~90°, … , 315°~360°). For each interval, we computed the average firing rate of the neuron at every corresponding time point. The resulting graph illustrates the firing rates for each angle and its subsequent $45°$ range. Furthermore, we applied a cosine function to fit the mean firing rates and visually represented it in the graph.

For each tuning curve, we used the thickness of the curve to reflect the data size; the thicker the curve, the more data points were included for that state. Additionally, we used the color of the curve to represent the performance of the neurons; the darker the color, the higher the R^2 of the neuron within that state.

Fig. 2b, 2c, 2d plot the directional tuning curves without the fitting step of four example neurons from 220324-S2, 220724-S3, and 220724-S3, respectively.

5 State-dependent decoding of handwriting trajectories

5.1 Particle-based estimation for the state-dependent decoder

The DyEnsemble model can be solved using a particle filtering algorithm, which estimates a posterior distribution with a set of particles and their associated weights. Specifically, to estimate $p(x_k|y_{0:k})$, two conditional probabilities of $p(x_k|h_k(\cdot) = \mathcal{H}_m(\cdot), y_{0:k})$ and $p(h_k(\cdot) = \mathcal{H}_m(\cdot)|y_{0:k})$ should be estimated.

Let's consider time step k, where we already have prior information $p(h_k(\cdot) = \mathcal{H}_m(\cdot)|y_{0:k-1}), m = 1, \dots, M$ and a set of particles with weights $\{\omega_{k-1}^i, \chi_{k-1}^i\}_{i=1}^{N_{\text{par}}}$, where N_{par} represents the size of the particle set. Here, χ_{k-1}^i denotes the i^{th} particle with weight ω_{k-1}^i at time $k-1$. The particles can be randomly initialized, or given by the kinematic data from the training set. We used the kinematic data of the training set in this study. The term of $p(x_k|h_k(\cdot) = \mathcal{H}_m(\cdot), y_{0:k})$ can be approximated by:

$$p(x_k|h_k(\cdot) = \mathcal{H}_m(\cdot), y_{0:k}) \approx \Sigma_{i=1}^{N_{\text{par}}} \omega_{m,k}^i \delta(x_k - \chi_k^i). \quad (12)$$

Here, $\delta(\cdot)$ represents the Dirac function, and the term $\omega_{m,k}^i$ is the normalized weight of particle χ_k^i when using the encoding model $\mathcal{H}_m(\cdot)$, and $\Sigma_{i=1}^{N_{\text{par}}} \omega_{m,k}^i = 1$.

The term of $p(h_k(\cdot) = \mathcal{H}_m(\cdot)|y_{0:k})$ can be recursively computed given $p(h_k(\cdot) = \mathcal{H}_m(\cdot)|y_{0:k-1})$ and $p_m(y_k|y_{0:k-1})$, and the particle-based approximation for $p_m(y_k|y_{0:k-1})$ is given by:

$$p_m(y_k|y_{0:k-1}) \approx \Sigma_{i=1}^{N_{\text{par}}} \omega_{m,k-1}^i p_m(y_k|\chi_k^i). \quad (13)$$

Here, $p_m(y_k|\chi_k^i)$ represents the Gaussian likelihood of particle χ_k^i when using the encoding model $\mathcal{H}_m(\cdot)$.

5.2 State prediction accuracy (Fig. S5a, S5b)

To verify the accuracy of the DyEnsemble state predictions, we conducted experiments on 22 sessions. For each session, we first applied three-fold cross-validation to divide it into training and test sets. After the division, we followed three steps:

1. State Identification on the Training Set Using TFC: We used the TFC method to identify states in the training set by inputting neural and writing signals, resulting in state segmentation

(corresponding to the first row in Fig. S5a).

2. State Identification on the Test Set Using TFC: Similarly, the TFC method was applied to the test set using the same inputs (neural and writing signals). The results from this step served as the ground truth for subsequent state predictions (corresponding to the second row in Fig. S5a).
3. State Prediction on the Test Set Using DyEnsemble: The DyEnsemble method was employed to predict states in the test set, this time using only neural signals without writing signals. DyEnsemble calculated the probability of each state based on the current neural signal input, converting these probabilities into state weights. In the figure, colors represent the dominant model at each time point, and the transparency indicates the weight of the dominant model (corresponding to the third row in Fig. S5a).

Fig. S5a is an example of one repeat from session 230327-S2. Fig. S5b shows the statistical results of all repeats across 22 sessions and $M = 10$ states (n = 800). $\text{Hard weight accuracy} = N_{\text{consist}}^{(m)} / N_{\text{GT}}^{(m)}$ is a metric used to measure the accuracy of state predictions, where $N_{\text{GT}}^{(m)}$ is the number of time bins where the m^{th} model is the ground truth, and $N_{\text{consist}}^{(m)}$ is the number of time bins where the m^{th} model is both the predicted dominant model and the ground truth.

Fig. S5b displays the distribution of the accuracy of DyEnsemble's state predictions (in purple) compared to the distribution of accuracy after shuffling the predicted state weights (in grey). The accuracy after shuffling is approximately at the basic level of a 10-class classification, around 1/10, which is significantly different from the accuracy of DyEnsemble's state predictions.

5.3 Visualization of decoded writing trajectories (Fig. 6b, S6a)

We selected the four Chinese characters “脑机接口” (meaning: Brain-computer interface) to illustrate the decoding performance in Fig. 6b. Two decoders were compared, with one state-dependent decoder (DyEnsemble, see Methods for details) and a decoder without considering states (Kalman filter). Initially, the spike counts were grouped into 200 ms bins using a sliding window with a step of 50 ms. To smooth the signals, a moving average with a window size of 10 bins was applied. For decoding, we employed a cross-validation approach known as leave-one-repeat-out. This means that the preprocessed data was divided into training sets and test sets according to the number of times each character was repeated. In each iteration, we used one repeat for test and the remaining repeats for training.

For the DyEnsemble decoder, we first utilized the TFC algorithm to identify $M = 10$ states and their associated models in the training data. The parameter l_{smooth} was set to 10, and the early-stopping threshold β was set to 0.001. Subsequently, we used the dynamic ensemble algorithm to calculate the weights for each model and predict the kinematic states based on the neural signals at each time step during testing. The particles used in the algorithm were unique kinematic states from the training set, and the sticking factor α was set to 0.1.

For the Kalman filter (KF), we employed a velocity Kalman filter with a linear-Gaussian state-space model:

$$x_k = Ax_{k-1} + b + \varsigma_{k-1} \tag{14}$$

$$y_k = Hx_k + \varepsilon_k \tag{15}$$

where k denotes the time step index, $x_k \in \mathbb{R}^{d_x}$ is the kinematic state and $d_x = 2$ is the dimensions of x_k, including the writing velocities in X-axis and Y-axis. $y_k \in \mathbb{R}^{d_y}$ is the neural measurement and d_y is the dimensions of y_k, determined by the number of neurons recorded in this session. $\varsigma_k \sim N(0, \sigma_\varsigma^2)$, $\varepsilon_k \sim N(0, \sigma_\varepsilon^2)$ are state transition noise and measurement noise.

The stroke trajectory was obtained by integrating the decoded velocities to visualize the characters. The start point of each stroke was predefined with the knowledge of the character, i.e., we applied the standard stroke start position with the Kai font of the character.

In Fig. S6a, we used data from the example session 230724-S3, which included four different types of writing content: Chinese characters, English letters, shapes, and numbers. Apart from the differences in writing content, all other decoding settings were consistent with those used in Fig. 6b.

5.4 Details of state-dependent decoding of handwriting (Fig. 6c, 6d, S6b)

To generate the statistical results for Fig. 6c, 6d, the decoding performance of the Kalman filter and the DyEnsemble decoder was evaluated using 18 experimental sessions, consisting of 1620 trials with 270 targets. For Fig. S6b, the decoding performance was assessed using one example session (230724S3), which included different types of writing content: Chinese characters, English letters, shapes, and numbers. Each writing type was decoded separately, using the leave-one-repeat-out test.

The results of Fig. 6c, 6d were offline decoding performance. The decoding was performed within each session using the three-fold cross-validation with no overlapping characters. With the training set, we first used the TFC to learn 10 encoding models, and these models served as the model pool for the DyEnsemble decoder. With the training data and the model pool, the DyEnsemble estimated the parameters for $f(\cdot)$ and $\varsigma_k \sim N(0, \sigma_\varsigma^2)$, and parameters of ε_{v_k}, using the estimation residuals of each encoding model in the pool. With these parameters, DyEnsemble can estimate the handwriting velocity recursively online, given the incoming neural signals.

We evaluated the decoding performance of the strokes in characters. Two evaluation metrics were used for the decoded velocity: root mean squared error (RMSE), and coefficient of determination (R^2). The formulas for these metrics are as follows:

$$\text{RMSE} = \sqrt{\frac{1}{T}\Sigma_{k=1}^{T}(x_k - \hat{x}_k)^2} \tag{16}$$

$$R^2 = 1 - \frac{\Sigma_k(x_k - \hat{x}_k)^2}{\Sigma_k(x_k - \bar{x})^2}. \tag{17}$$

In these formulas, x_k represents the velocity at time step $k \in [1, T]$, $\hat{x}_k$ represents the velocity the decoder estimates at time step k, and $\bar{x}$ is the average velocity of all time steps.

To calculate the final performance of each session, the metrics were averaged over all replicates. In the graph, each data point represents the decoding result of a session (n = 18). To assess the

significance of the decoding performance between the two algorithms, a paired *t*-test was performed, with $P < 10^{-4}$ in terms of RMSE and R^2.

5.5 Decoding performance with SUA and MUA (Fig. S5c)

To assess whether spike sorting step affects the results, we compared the decoding performance between SUA- and MUA-based decoders. In Fig. S5c, the decoding performance (R^2) of both state-independent (Kalman filter) and state-dependent (DyEnsemble) decoders is shown for SUA and MUA data. The dataset and testing protocol were identical to those used for Fig. 6c and 6d.

5.6 Cross session/day decoding performance (Fig. S5d)

To investigate the stability of neural activity over time, we evaluated the decoding performance across sessions spanning one month. Due to fluctuations in the number of recorded neurons across different days, we utilized MUA signals for decoding. In Fig. S5d, the matrix shows the cross-session decoding performance (R^2) using the DyEnsemble decoder. Each element in the performance matrix at position (*i, j*) represents the decoding accuracy when training the model using neural and kinematic signals from session *i*, and testing it on neural signals from session *j*.

5.7 Online decoding settings and delay analysis

We carried out online experiments that decoded the neural signals into handwriting trajectory with the state-dependent decoder in real-time and output the decoded handwriting trajectory to a robotic arm to evaluate the possibility of online handwriting BCIs.

In the online decoding process, neural signals were unsorted, and we used the thresholding approach to detect spike events for each channel. The time bin was 10 ms without overlap for quick response. The online experiment contains a training session and a test session. The training session used the 'sentence writing with visual guidance paradigm' (SW) without output to collect several sentences to train the state-dependent decoder, and then the decoder was deployed for the online decoding process in the second session (SW paradigm, with the robotic arm output). For the demonstration in Supplementary Video 2, the training session was 230327-S1 and the test session was 230327-S2. The detailed training and test contents are listed in Supplementary Table 3. The online decoding R^2 was 0.51 for session 230327-S2.

Supplementary Table 3. Training and test sessions for online demonstration

Session	N_C	N_R	Characters	P
230327-S1	30	3	今天很开心我在写字浙江大学脑机接口你好他乡还吗明见外面气再	SW
230327-S2	8	5	浙江大学脑机接口	SW

The robotic arm was controlled by position signals (x, y, and z) at each 10 ms, to hold a pen and

write on a whiteboard. Chinese characters contain multiple strokes such that there were pen-lift and pen movements between adjacent strokes. As the verification of the state-dependent encoding model, we only decoded the movements of the stroke writing process and output the trajectories of strokes. Thus, extra information was required to composite strokes into characters: 1) the start and end timestamp of each stroke, and 2) the start position of each stroke on the board. In the proof-of-concept demonstration, to know when each stroke started and ended, for each character, we adopted the timestamps of the start and end for each stroke in the generated trajectory (synchronous to the instruction video). The start position of each stroke was predefined according to the Kai font of the character. During the online decoding process, at the end point of each stroke, the robotic arm automatically lifted the pen and moved the pen tip to the start point of the next stroke.

The computation of neural decoding could be achieved in each 10 ms bins. The computational cost for the state-dependent decoder was 11.8±3E-3 ms per bin with CPU only, and 4.15±2E-4 ms per bin with GPU acceleration. We used GPU acceleration for online experiments. The workstation used for computing was equipped with an AMD CPU (AMD Ryzen 5 5600G with Radeon Graphics, 3.90 GHz) and an NVIDIA GPU (NVIDIA GTX 1080), with 64G of memory.

During the online process, we observed a delay in robotic arm control. The delay was mostly due to the movement of robotic arm between the writing of adjacent strokes. The first delay was caused by the pen-tip lifting process, where after writing each stroke, the robotic arm should pull the pen tip 2 mm away from the whiteboard, move the pen to the start point of the next stroke, and then push the pen tip to the whiteboard again. The pull and push process was additional for the robotic arm, such that it caused a delay (about 300 ms per character). The second delay was the movement of robotic arm from the end of the preceding strokes to the start of the succeeding strokes, which can be uncertain, since the end points of the preceding strokes relied highly on the online decoding results.