Published in final edited form as:
J Neural Eng. 2008 December ; 5(4): 455–476. doi:10.1088/1741-2560/5/4/010.

Neural control of computer cursor velocity by decoding motor cortical spiking activity in humans with tetraplegia*

Sung-Phil Kim[1,6], **John D Simeral**[2,3], **Leigh R Hochberg**[2,3,4], **John P Donoghue**[2,3,5], and **Michael J Black**[1,6]

[1] Department of Computer Science, Brown University, Box 1910, 115 Waterman St, Providence, RI 02912, USA

[2] The Center for Restorative and Regenerative Medicine, Rehabilitation R&D Service, Department of Veterans Affairs, Providence, RI 02912, USA

[3] Department of Neuroscience, Brown University, Providence, RI 02912, USA

[4] Department of Neurology, Massachusetts General, Brigham and Women's and Spaulding Rehab. Hospitals, Harvard Medical School, Boston, MA 02114, USA

[5] Cyberkinetics Neurotechnology Systems, Inc., Foxborough, MA 02035, USA

Abstract

Computer-mediated connections between human motor cortical neurons and assistive devices promise to improve or restore lost function in people with paralysis. Recently, a pilot clinical study of an intracortical neural interface system demonstrated that a tetraplegic human was able to obtain continuous two-dimensional control of a computer cursor using neural activity recorded from his motor cortex. This control, however, was not sufficiently accurate for reliable use in many common computer control tasks. Here, we studied several central design choices for such a system including the kinematic representation for cursor movement, the decoding method that translates neuronal ensemble spiking activity into a control signal and the cursor control task used during training for optimizing the parameters of the decoding method. In two tetraplegic participants, we found that controlling a cursor's velocity resulted in more accurate closed-loop control than controlling its position directly and that cursor velocity control was achieved more rapidly than position control. Control quality was further improved over conventional linear filters by using a probabilistic method, the Kalman filter, to decode human motor cortical activity. Performance assessment based on standard metrics used for the evaluation of a wide range of pointing devices demonstrated significantly improved cursor control with velocity rather than position decoding.

1. Introduction

Injury or disease affecting subcortical, brainstem, spinal or neuromuscular motor pathways, can result in paralysis while leaving cerebral function intact. A neural interface system (NIS) is a brain–computer interface that provides alternative connections between neurons and assistive devices and thus has the potential to restore lost function. Several studies in able-bodied non-human primates have demonstrated that extracellular neural activity from a

*Disclosure. JPD is the Chief Scientific Officer and a director of Cyberkinetics Neurotechnology Systems (CYKN); he holds stock and receives compensation. JDS has been a consultant for CYKN. LRH receives clinical trial support from CYKN.

[6] Authors to whom any correspondence should be addressed. spkim@cs.brown.edu and black@cs.brown.edu.

population of neurons in the motor areas of cerebral cortex can be converted into a continuous control signal for the operation of computers or robotic devices [1–8] (see [9–12] for review). Central to the function of an NIS is an algorithm for decoding neural activity into a stable, reliable control signal. In a recent study, Hochberg *et al* [13] demonstrated that paralyzed humans could operate an NIS that translated motor cortical neural activity into a two-dimensional (2D) computer cursor position. That report found that neural signals related to movement remained in the arm area of human motor cortex years after spinal cord injury and that these signals could be activated by imagined or attempted movement, essential requirements for the operation of an NIS. Two-dimensional (2D) cursor control, along with prosthetic hand and robotic arm control, was derived using a linear filter decoding method that computed 2D cursor position from a linear combination of neural population firing rates over a short-time history [14].

While demonstrating the ability for a tetraplegic human to control a neural cursor well enough to operate a simplified computer interface, the quality of cursor control reported in the Hochberg *et al* study [13] was typically not at the level achieved by able-bodied users of standard computer pointing devices. In particular, the cursor trajectories were longer and more curved than the typical straight-line movements of able-bodied users. Further, it was difficult for the user to stop the cursor at a target location and to maintain a fixed position. Our goal is to improve NIS function while shedding light on key issues in the design of any NIS including: (1) the features of intended movement (e.g. kinematic representation) that are most natural for cursor control; (2) how training of the neural decoding algorithm affects cursor control accuracy; (3) the statistical stationarity of neural tuning properties between training and testing; (4) quantification of the control accuracy obtained using different features and decoding algorithms (in terms of cursor trajectory accuracy and target acquisition rate); and (5) the effect of the choice of decoding algorithm on performance. For two participants in our ongoing pilot clinical trial, investigation of each of these issues is described below. A central question addressed here is whether cursor position or velocity provides a more natural movement ‘feature’ for accurate neural cursor control. Specifically, we address this question in the context of closed-loop neural cursor control using the motor cortical population activity generated by the intended (as opposed to ‘performed’) actions of humans with tetraplegia.

Numerous studies have shown that neuronal spiking activity in the primary motor cortex (MI) is correlated with arm kinematic and dynamic parameters including limb forces, joint torques, hand direction, hand speed and hand position [14–22]. Hand position and velocity have both been used (together and separately) for closed-loop neural cursor control in able-bodied non-human primates using a variety of decoding algorithms including the population vector method [5], linear filtering [1–4,23] and Kalman filtering [24,25]. Whether cursor position or velocity is more appropriate for neural control by a tetraplegic human has not been firmly established (see [26] for a recent offline analysis). Understanding the implications of this choice of kinematic representation on cursor control performance is critical for the development of a useful NIS.

Many previous closed-loop cursor control experiments in monkeys relied on training data containing simultaneously recorded hand movements and neural activity to train neural decoding algorithms. In a human with tetraplegia, training of the NIS must be achieved in the absence of physical movement. Hochberg *et al* [13] proposed a combination of open-loop and closed-loop filter training that enabled subjects to gain control of cursor position. We extended this training procedure to enable the control of cursor velocity.

In the present study, in two tetraplegic humans, we compared the neural cursor control performance of two decoding approaches based on position or velocity control with two

different decoding algorithms. We first quantified the position and velocity tuning of motor cortical neurons during imagined cursor movement. We then quantified closed-loop neural cursor control with both position and velocity decoding using metrics defined in the ISO 9241-9 standard for evaluating pointing device performance [27]. Additionally, we measured other relevant aspects of cursor control and movement including deviation from the desired trajectory, movement direction changes and movement variability [28] in order to provide a comprehensive set of measures by which control can be compared across studies, across clinical disorders and across decoding approaches. Finally, we evaluated two decoding algorithms to test the relative importance of the kinematic representation versus the specific decoder. Here, we compared the linear filter [29] which was used in Hochberg *et al* [13] and the Kalman filter [30] which showed good performance in previous non-human primate studies [24,25].

It is also worth noting that the experiments reported here show the feasibility of neural cursor control by a paralyzed human using an intracortical NIS more than 1 year post implant.

2. Methods

2.1. Participants

Clinical trial sessions[7] of the BrainGate NIS (Cyberkinetics Neurotechnology Systems, Inc. (CYKN)) were conducted by Cyberkinetics technicians with two participants with tetraplegia (paralysis of both arms and both legs). Participant S3 is a 54 year old woman who had thrombosis of the basilar artery and extensive pontine infarction 9 years prior to trial recruitment. Participant A1 was a 37 year old man with amyotrophic lateral sclerosis (ALS, motor neuron disease), recruited to the trial 6 years after being diagnosed with ALS. Both participants were right-hand dominant, and the intracortical array was placed in the left precentral gyrus in the region of the arm representation [13].

2.2. Recording

During the sessions, neural signals were recorded from the motor cortex of the participants using a chronically-implanted 96-channel Cyberkinetics microelectrode array and the BrainGate NIS. After digitization (30 kKz per channel) real-time, amplitude-thresholding software (see Suner *et al* [31] for details) was utilized to discriminate different waveshapes on each channel. Putative single neurons and apparent multi-neuron activity with consistent waveforms [31] (both referred to here as ‘units’) were accepted or rejected for inclusion in the study at the beginning of each session based on visual inspection of the isolated waveforms with no further criteria applied to identify single neurons. We did not analyze the nature of these units in detail but some were clearly defined single units, some contained multiple cells with waveforms that could not be confidently segregated from each other and some were low-amplitude intermixed signals that modulated with the task. During recording, subjects viewed a computer monitor that displayed task information related to various cursor control tasks as described below.

We refer to each session with the notation: ⟨participant⟩—⟨days since implant⟩; for example S3-40 is the session for participant S3 on day 40 after implantation. Sessions analyzed in the present study fell into one of three categories based on the type of training task and the decoding algorithm, referred to as ⟨kinematic model⟩ ⟨decoding method⟩ ⟨task⟩. PLP

[7]A pilot clinical study of the BrainGate Neural Interface System was initiated by Cyberkinetics Neurotechnology Systems, Inc. under a Food and Drug Administration (FDA) Investigational Device Exemption (IDE) and with Institutional Review Board (IRB) approvals; the studies began in May 2004.

sessions tested position decoding using a linear filter trained with a pursuit tracking task; VKC sessions tested velocity decoding using a Kalman filter trained with the center out task; and VLKC sessions tested velocity decoding using linear and Kalman filters trained on the same day with the center out task. The various tasks are described in detail below. The analyses here are based on all complete sessions between 40 and 418 days after implant in S3 and 85 to 224 days after implant in A1 for which the decoding and cursor control evaluation methods studied here were used.

2.3. Training

2.3.1. Training procedure—A main purpose of 'training' in our clinical study was to set the parameters of the decoding algorithms such that they define an optimal mapping between neural activity and cursor control signals. A training procedure was developed previously (see Hochberg *et al* [13]) to simultaneously obtain neural activity patterns and cursor kinematics to enable the estimation of the decoding algorithm parameters without accompanying limb movements. In this procedure, a training cursor (TC) was displayed on a computer monitor and moved to generate cursor trajectories (details are given below). During this presentation, the participants were instructed to imagine moving their arm or hand as if they were controlling the TC. Neural signals recorded during this imagined movement together with the TC kinematics (position or velocity) were used to train the decoding algorithms.

The overall training procedure was composed of a series of short recording periods, called 'blocks', each of which lasted 1–1.5 min. We devised two types of training blocks: open-loop (OL) and closed-loop (CL) blocks. In OL blocks, the TC and the target were shown together on the monitor. The number of OL blocks varied over session groups: four 1 min OL blocks for PLP sessions, two (mean 2, SD ± 1) 1.2 min blocks for VKC sessions and three 1.5 min blocks for VLKC sessions. We used the data from the OL blocks to train an initial decoding algorithm.

The OL blocks were followed by several closed-loop (CL) blocks in which a feedback cursor (FC) was simultaneously displayed on the monitor with the TC. The FC motion was determined using the decoding algorithm trained from the OL trials. The presentation of the FC provided additional information through visual feedback to the subjects about how well their intended actions were being decoded. FC presentation was motivated by an assumption that the participants might adjust their neural activity to improve cursor control by seeing the error between the FC and TC. The decoding algorithm was re-trained at the end of every other CL block to incrementally update the parameters. Four 1 min CL blocks were used in PLP sessions, six (mean 6, SD ± 2) 1.2 min blocks in VKC sessions and four 1.5 min blocks in VLKC sessions. See figure 1(*a*) for an illustration of the procedure of OL and CL training.

In PLP recording sessions, the TC was moved manually by the technician. In VKC and VLKC sessions, the TC was moved by a computer-generated bell-shaped velocity profile (figure 1(*b*)) to provide the participants with well-defined velocity training patterns. The participants rested between blocks for a short period (~1 min on average).

2.3.2. Cursor movement tasks for training decoding algorithms—Two different training tasks were used depending on the representation of cursor kinematics (position or velocity): position-based decoders were trained using a random target pursuit-tracking task [13,14] while velocity-based decoders were trained using a center-out-and-back (to center) task [32,33]. Note that during training a TC was always present on the screen. This made these training tasks more similar to pursuit tracking than step tracking. However, during

testing, only the NC was present under closed-loop control and the tasks can be seen as step-tracking tasks.

In the random pursuit-tracking task, the TC moved from a starting location toward a target that was randomly placed on the screen. When the TC intersected the target, an audio feedback cue was provided and the next target immediately appeared at another random location, keeping only one target at a time on the monitor. This task had the important property of spanning much of the space of screen positions, which provided the decoder learning method with a wide range of cursor position samples (see figure 1(*c*) for example of TC positions). The target and the TC occupied 90 × 90 pixels (visual angle: 3.8°) and 60 × 60 pixels (2.5°) of the screen, respectively, where the total screen size was 800 × 600 pixels (17 inch monitor). The distance from the participants' eyes to the center of the monitor was approximately 59 cm and the visual angle of the workspace on the screen was approximately 34.08°. The size of the FC during CL blocks occupied 102 × 126 pixels (4.3° × 5.3°).

In the center-out-back task, four peripheral targets (0°, 90°, 180° and 270°) and one center target were displayed on the screen. The TC started at the center target, holding there for 1.2 s, and then began moving to one of four peripheral targets which was highlighted. When the TC reached the target it remained there for 0.5 s before tracing its path back to the center target. The point-to-point TC movement followed a bell-shaped computer-generated speed profile (figure 1(*b*)) that spanned T seconds. The reaching time, T, varied across the session in the range [1.5 s, 4.5 s]. A single center-out-back TC trial took $(1.7 + 2T)$ seconds. The target location was pseudo-randomly selected by a predetermined program. We used the center-out-back task instead of the standard *center-out* task with re-centering. In this latter task, a cursor moves smoothly from the center to peripheral targets but then jumps automatically back to the center target. This procedure causes cursor position and velocity to be highly correlated with each other. In contrast, the center-out-back task enables us to sample two opposite cursor velocities for each cursor position. This makes it easier to uncover whether a modulated neural activity is related to position and/or velocity. In the earlier sessions including S3-{254, 261, 280, 282} and A1-{197, 216}, the target, the TC and the FC occupied 90 90 (3.8°, 60 × 60 (2.5°) and 60 × 60 (2.5°) pixels, respectively. The distance from the center to the target was 210 pixels (8.9°). In the later sessions including S3-{285, 287, 408, 412, 418} and A1-224, the sizes of the target, the TC and the FC were reduced to 48 × 48 (2.0°), 40 × 40 (1.3°) and 40 × 40 (1.3°) pixels, respectively. The distance from center was increased to 255 pixels (10.8°) for vertical targets and 300 (12.7°) for horizontal targets to more completely cover the screen space.

2.4. Neural cursor control testing tasks

A closed-loop center-out-back task was used to evaluate neural cursor control performance. This testing task shared a basic structure with the center-out-back training task, however it differed in several respects (see below). The NC was not automatically re-centered, making this task closer to normal continuous computer mouse activities than the standard, recentering, center-out task [34].

The center-out-back testing task used in PLP and VKC sessions was the same as the one used in Hochberg *et al* [13]. A target (120 × 120 pixels; visual angle: 5.1°) was positioned at one of four fixed locations (0°, 90°, 180°, 270°) with the distance of 210 pixels (8.9°) from the center and the participant was asked to control the NC (60 × 60 pixels; 2.5°) to acquire the target. If the NC reached the target and dwelled on it longer than a preset period (500 ms), the target was deemed to be acquired and audio and visual feedback was given for 1s. If a timeout period (7 s) elapsed before target acquisition, another audio feedback cue was provided indicating failed target acquisition (with no change of the target icon). After target acquisition or failure due to timeout, the target was relocated to the screen center and the

participant had to move the NC to acquire it before starting the next target acquisition trial. There was no time limit for this center-target acquisition. Data collected during center-target acquisition is not analyzed in this study.

In VLKC sessions, a more challenging center-out-back testing task was used. There were eight peripheral targets and one center target presented on the screen. Smaller targets were located further from the center: the sizes of the targets and the NC were 48 × 48 (2.0°) and 40 × 40 (1.7°) pixels, respectively, and the distances from center to the targets were 255 pixels (10.8°) for vertical direction and 300 (12.7°) for horizontal direction, respectively. A trial could end without success not only when the timeout period was exceeded but also when a false target was acquired. The timeout period was 10 s for this task. Compared to the above four-target testing task, this task increased the index of task difficulty (ID) [27] by 2.5 times, increased the number of targets, and added another failure condition by allowing the potential selection of false targets.

2.5. Decoding algorithms

Decoding is the process of converting recorded neural signals (e.g. firing rates) into cursor movement. In this study, we compared two decoding algorithms: linear filtering versus Kalman filtering.

Let $\mathbf{x}_t$ be a $D \times 1$ vector of cursor kinematics such as position $[p_x, p_y]^T$ or velocity $[v_x, v_y]^T$ in the x (horizontal) or y (vertical) direction sampled at a discrete time instant t. $D = 2$ in this study as we decode the 2D cursor kinematics, yet the decoding algorithms presented here can be extended for the cases with $D > 2$. Note that decoding velocity (i.e. $\mathbf{x}_t = [v_x, v_y]^T$) is equivalent to decoding direction and speed in the polar space. Let $\mathbf{z}_t$ be an $N \times 1$ vector of firing rates of N units observed at t. Firing rates for each unit were approximated using spike counts computed in fixed, non-overlapping, time bins. The bin size was set as either 50 ms for PLP sessions or 100 ms for the remainder of the sessions. Decoding algorithms establish a causal relationship between these neural firing rates and the cursor kinematics such as position or velocity.

2.5.1. Linear filter—Using the linear filter, each dimension d of the current kinematic vector $\mathbf{x}_t$ was predicted separately using a linear combination of the current and past firing rates, $\mathbf{z}_t$, $\mathbf{z}_t-1$,..., $\mathbf{z}_{t-L+1}$,

$$x_t^d = b_d + \sum_{i=0}^{L-1} \mathbf{w}_{i,d}^T \mathbf{z}_{t-i} + \varepsilon_t, \qquad \text{for} \quad d = 1, \ldots, D, \tag{1}$$

where L denotes the length of the firing history, ε_t is a random variable assumed to be white Gaussian noise with zero mean, b_d is a scalar bias term for dimension d and $\mathbf{w}_{i,d}$ is an $N \times 1$ vector of filter coefficients for dimension d and $\mathbf{z}_{t-i}$. Motivated by previous intracortical brain–computer interface (BCI) studies that took L to be approximately a 1 s history of firing rates [1–4], here L was chosen to be 20 (bins) when 50 ms bins were used or 10 (bins) when 100 ms bins were used. The optimal linear weights $\widehat{\mathbf{w}}_d = \left[\mathbf{w}_{0,d}^T, \ldots, \mathbf{w}_{L-1,d}^T\right]^T$ were obtained using least-squares regression to minimize the mean-squared error between estimated and training cursor kinematics.

2.5.2. Kalman filter—The Kalman filter [35] is a Bayesian inference algorithm that recursively infers the cursor kinematics from the history of firing rates. In this formulation, inference of kinematic signals $\mathbf{x}_t$ conditioned on the whole observation of firing rates with a

possible time offset j, $\mathbf{z}_{1:t-j} = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_{t-j}\}$, is performed through recursive Bayesian estimation [25],

$$p\left(\mathbf{x}_t|\mathbf{z}_{1:t-j}\right) + \frac{1}{\kappa} p\left(\mathbf{z}_{t-j}|\mathbf{x}_t\right) \times \int p\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right) p\left(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1-j}\right) \mathrm{d}\mathbf{x}_{t-1}, \tag{2}$$

where k is a normalization constant. In this estimation, the *a posteriori* probability $p(\mathbf{x}_t \mid \mathbf{z}_{1:t-j})$ at time t is inferred by updating the previous estimate $p(\mathbf{x}_{t-1} \mid \mathbf{z}1:_t-1-_j)$ at $t - 1$ using a system model $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}$ and an observation model $p(\mathbf{z}_{t-j} \mid \mathbf{x}_t)$ with neural data $\mathbf{z}_{t-j}$. The observation model describes how the observed firing rates at $t - j$ (with a possibly nonzero lag offset j) are generated from kinematics $\mathbf{x}_t$ and the system model describes how the cursor kinematics evolve from $t - 1$ to t. In the Kalman filtering algorithm, the observation model and the system model are approximated as linear Gaussian models,

$$\begin{aligned} \mathbf{z}_{t-j} &= \mathbf{H}\mathbf{x}_t + \mathbf{q}_t, & \mathbf{q}_t &\sim G(0, \mathbf{Q}) \\ \mathbf{x}_t &= \mathbf{A}\mathbf{x}_{t-1} + \omega, & \omega_t &\sim G(0, \Omega), \end{aligned} \tag{3}$$

where **H** and **A** are time-invariant linear coefficient matrices for each model and $\mathbf{q}_t$ and $\boldsymbol{\omega}_t$ are random vectors distributed with zero-mean Gaussian noise with covariance matrices **Q** and **Ω**, respectively. $G(\cdot)$ here denotes a multi-variate Gaussian distribution. Note that $\mathbf{z}_t$ and $\mathbf{x}_t$ are centered to be zero-mean vectors. The parameters **H, A, Q** and **Ω** are estimated from training data using a least-squares method. Given the learned model, decoded kinematics can be inferred at every time instance using a closed-form recursion. Details of the model training and use of the Kalman filter for neural decoding can be found in Wu *et al* [24,25].

It is worth noting that both the linear filter and the Kalman filter integrate the neural signals over time. The linear filter achieves this by explicitly building a finite impulse response (FIR) filter with a 1 s memory of the firing rate history where the FIR filter tends to pass the low-frequency signal. The Kalman filter does this by the recursive equations, integrating the information from the history of neural signals (see [25] for comparison between two filters). A fundamental difference between the filters is that the Kalman filter is a Bayesian model that, in addition to modeling the relationship between the neural and movement signals, also incorporates a prior model of the cursor movement.

2.6. Correlation analysis during open-loop training

To evaluate the relationship between the firing rates and the TC kinematic parameters independent of a specific decoding method, we computed the Pearson correlation coefficient (CC) [36] between the firing rate for each unit and the TC kinematic parameters during the OL training blocks collected from multiple sessions where either the random pursuit tracking or the center-out-back task was used. In our analysis, each unit's firing rate was smoothed by a low-pass filter with a unit gain and a cutoff frequency at $0.2 \times f_s$. Here, f_s represents a sampling frequency for the firing rate and is set to 10 Hz (corresponding to the 100 ms bin width). To account for latency between the TC movement and motor cortical activity, we computed the CC for each unit at time lags from j = 0 s, 0.1 s,..., 2 s where the lag j means the firing rate leads the kinematic signals by j seconds. We empirically searched for the optimal lag at which the CC was maximized. Note that we only explored non-negative lags to maintain a causal relationship between firing and movement. This is necessary as neural activity must precede action for effective closed-loop cursor control.

Specifically, let z^i_{t-j} be the smoothed firing rate of unit i at the time instant $t - j$ and x^d_t be the dth dimension of the kinematic parameter vector $\mathbf{x}_t$. The CC between z^i_{t-j} and x^d_t was estimated from data using

$$CC(i,d) = \frac{E\left[\left(z^i_{t-j} - \bar{z}^i\right)\left(x^d_t - \bar{x}^d\right)\right]}{s_z s_d}, \quad (4)$$

where $\bar{z}$ and s_z (respectively, $\bar{x}$ and s_d) are the mean and standard deviation of z^i_{t-j} (or x^d_t), and $E[\cdot]$ represents the expectation. Further, we used the absolute value such that $|CC(i, d)| = 0$ represents no correlation and $|CC(i, d)| = 1$ corresponds to perfect correlation (including perfect inverse correlation). The significance of the correlation for each unit was evaluated using a t-test (with significance determined at the 1% level). If a unit showed significant correlation with this test for at least one dimension of $\mathbf{x}_t$ at any lag j, then the unit was considered to be significantly correlated with $\mathbf{x}_t$.

2.7. Decoding accuracy during closed-loop training

To evaluate cursor decoding performance during CL training blocks we computed the CC between the TC and the FC kinematics in each block. Since the number of CL blocks varied across sessions, we evaluated only the first four CL blocks collected from each session. After finding an optimal lag as above, the CCs for each of the vertical and horizontal kinematic parameters were averaged. We used Fisher's transform [37] to estimate a 95% confidence interval for the CC estimated in each block.

2.8. Evolution of directional tuning

Velocity-based decoding specifically exploits the directional tuning of motor cortical units that persists in humans with tetraplegia [13,26]. To investigate the consistency of this directional tuning we analyzed whether the directional tuning of individual units changed between training and testing epochs. This analysis was applied to the recording sessions where velocity-based decoding was performed (six sessions for S3 and three sessions for A1). In this analysis, we used a cosine tuning function [17] to fit the firing rates as a function of cursor direction. This approach has been widely used for studying neural tuning to arm/hand direction in non-human primates. For training epochs, we computed the tuning function using the TC direction. For testing epochs, where the TC was not present, we computed the tuning function with the *intended* direction that was defined as a direction from the current position of the NC to the designated target (see [26] for introduction of the intended direction). Using the cosine tuning function fit to the firing rates of individual units, we computed the preferred direction (PD) and the tuning depth (TD) for each unit. The PD is a direction at which the fitted firing rate is maximal and the TD is the difference between the maximum and minimum values of the fitted firing rate. The confidence intervals of the estimated PD and TD were obtained with a bootstrap algorithm (bias corrected percentile method, 10^3 bootstrap re-samples) [38]. The PD (or TD) of each unit was considered to have changed significantly between training and testing epochs if the 95% confidence interval of the PD (or TD) estimated during testing did not overlap the 95% interval estimated during training.

2.9. Performance evaluation measures

To quantify NC control performance, we applied performance measures used in the field of human computer interaction (HCI) to evaluate pointing devices. The measures are composed

of two categories, including gross measures that evaluate the overall performance of a given task as defined in the ISO9241-9 standard [27], and more detailed measures that evaluate the spatio-temporal properties of continuous point-to-point movements [28].

Of the gross measures, we evaluated speed and accuracy of cursor control by computing mean movement time (MT) and error rate (ER) in target acquisition. MT was measured as the average time it took to reach and dwell on the target from a starting point. Here we measured MT for the trials only when the target was acquired. ER was defined as a percentage of the trials in which a designated target was not acquired. Note that in this study we do not report an information-theoretic throughput measure standardized by ISO9241-9 since it requires tasks with a range of task difficulties not present in our tests.

We also computed four fine performance measures adopted from MacKenzie *et al* [28], including orthogonal direction change (ODC), movement direction change (MDC), movement error (ME) and movement variability (MV); these are illustrated in figure 2. Given a task axis defined as a straight line connecting the starting position to the target position, ODC measures how consistently the cursor moves toward a target by counting direction changes orthogonal to the task axis and MDC measures the straightness of the NC path by counting direction changes parallel to the task axis. ME measures how much the NC path deviates from the ideal straight line (i.e. task axis) by computing the average deviation of NC positions from the task axis,

$$\mathrm{ME}=\frac{1}{m}\sum_{i=1}^{m}|y_i|, \tag{5}$$

where y_i is a distance (positive or negative, see figure 2) from the ith sample point on the NC path to the task axis and m is the number of NC samples from the starting point to the target. MV measures the variability of the NC path by computing the standard deviation of y_i,

$$\mathrm{MV}=\sqrt{\frac{\sum_{i=1}^{m}\left(y_i-\bar{y}\right)^2}{m-1}}. \tag{6}$$

2.10. Chance performance

We further evaluated neural cursor control performance in terms of decoding accuracy and target acquisition rates relative to chance as described below. These evaluations use estimates of chance performance and are dependent upon details of the task configuration and the decoding algorithm. Here we present methods of estimating two different chance performance measures: chance decoding accuracy during training and chance target acquisition rate during testing.

The chance decoding accuracy during training was estimated by training and testing a decoder with shuffled data in which neural signals and the TC kinematics were shuffled to be uncorrelated. To create surrogate data, we first randomly selected K (>2) training (OL or CL) blocks from a single recording session. Then, we replaced the TC data of the first $K-1$ blocks with different TC data that were randomly selected from different recording sessions in which the same training task was used. A decoder was trained using the data in these $K-1$ blocks. The trained decoder was then used to decode the TC kinematics from neural signals for the Kth block. The CC between the estimated and the true TC kinematics in this testing block was measured. This procedure was repeated 10^3 times to obtain an average

chance level for the CC. We estimated the chance level for each pair of the decoder and the associated training task (i.e. the position-based linear filter with random pursuit-tracking or the velocity-based Kalman filter with center out back).

We also estimated chance performance for the closed-loop center-out-back target acquisition task (four or eight peripheral targets). In an off-line analysis, we characterized the chance target acquisition rate by estimating the probability that the target would have been acquired by accident if the participant were not aware of the actual target location but only knew that it should be one of four (or eight) possible target locations. Again, target selection was determined by whether the NC dwelled on a specific target for 500 ms or more. If the NC were perfectly controlled, the *baseline chance rate* would be 25% for four targets (12.5% for eight). Depending on the characteristics of the NC decoding method, however, this baseline chance rate may not reflect the 'true' rate of selecting a target by accident. If neural control produced straight but not perfect trajectories, the chance rate would likely be lower by an amount proportional to the target acquisition performance; for instance, if NC control with some decoding algorithm resulted in acquisition of 80% targets, the baseline chance rate would be adjusted to 20%. If, instead, the decoding algorithm yielded 'unsteady' NC movements that could span a wide range of the computer screen, the NC could intersect and dwell on any of the target locations accidentally. This could increase the chance rate for a particular decoder above the baseline; this rate can be computed by simulation. Any decoder-specific changes to the baseline chance rate result in what we call the *adjusted chance rate*. In this case, the difference likely reflects the effect of unsteady NC behavior on target acquisition performance. Below 'chance rate' is used to mean the adjusted chance rate.

In this *post-hoc* analysis, we ran a computer simulation using the data recorded during performance of the target acquisition task. For each 'simulated' trial, we used a true NC trajectory with a randomly placed target at one of four or eight possible locations; in this way, we decoupled the observed NC trajectory from the original goal. The chance rate was computed as the percentage of the trials when the NC moved to the random target and dwelled on it for 500 ms within the timeout period. For each decoder, we repeated this procedure 10^3 times to obtain the average chance rate. The simulated data were collected from specific session datasets for different decoders (see table 1): five PLP sessions for the position-based linear filter with four targets (total 400 target acquisition trials); nine VKC sessions for the velocity-based Kalman filter with four targets (521 trials); and three VLKC sessions for the velocity-based linear and Kalman filters with eight targets (90 trials with the linear filter and 102 with the Kalman filter, respectively).

The method defined above results in a chance rate that is generally higher than that used in Hochberg *et al* [13]. For consistency and comparison with that earlier study we also measured the *control chance rate* (control rate) for the four-target tasks (see [13] for details). In this off-line analysis, both the correct target and a *single* false target (randomly selected from the three other targets) were placed on screen during each *post-hoc* trial. We measured the percentage of the trials in which the NC passed through (and dwelled on) the false target by accident before it reached the correct target. Note that we did not have to perform this off-line analysis for VLKC sessions since, for these sessions, all eight targets were shown simultaneously and a false target could be selected on-line if the NC dwelled on it; thus, this is already included in the error rate.

Finally, we define a significant target acquisition rate (STAR) as the difference between the observed target acquisition rate and the control rate. This STAR quantifies what portion of the observed target acquisition rate is indeed achieved by intentionally moving the NC to the

target. STAR is useful to compare cursor control performance when the control rate varies across different decoders on identical tasks.

3. Results

We investigated the impact of several NIS design choices on neural cursor control by two participants with tetraplegia (S3 and A1). Both participants obtained neural control of a computer cursor during the performance of a four-direction radial target acquisition task and results are reported for 17 sessions ranging from 40 to 418 days after implantation of the array in which the center-out-back task was used to assess the performance (with variations as described in section 2). Neural activity was observed in all these sessions and the number of spiking units detected per session varied from 13 to 179 (mean 63.8, SD 61.9, 12 sessions) for S3 and from 85 to 117 (mean 95.4, SD 18.2, 5 sessions) for A1, respectively. The units were isolated by utilizing signal processing and spike sorting methods applied previously in this ongoing trial [13].

In several experiments below we explored five important issues relevant to developing an NIS: (1) tuning of neuronal units to cursor position and velocity during OL training; (2) the effects of OL and CL training on acquisition of cursor control with position or velocity; (3) the stability of directional tuning between training and testing; (4) the closed-loop performance of position-based cursor decoding using the linear filter versus velocity-based decoding using the Kalman filter; (5) The importance of the kinematic representation versus the decoding algorithm for cursor control.

3.1. Kinematic correlation analysis

The CC between the firing rate and the kinematic parameters was measured for every unit recorded during the OL training blocks where the participants imagined arm/hand movements following the TC motion. During OL training neural activity should be related to the properties of any imagined movement (or, perhaps, the observed visual stimulus) and is independent of any particular decoding algorithm since no NC is presented. The analysis was performed with 1226 units recorded in both participants recorded over 17 recording sessions.

We first computed the optimal lag for each unit at which the CC was maximized. The distribution of the optimal lag over the neural population is shown in figure 3(*a*) and reveals that for most units the CC was maximized at zero lag for both position and velocity. Based on this analysis we assume zero lag for all the Kalman filter decoders used in this study (namely, $j = 0$ in equation (2)).

From the statistics of the CC measures, we found that the firing rates of the majority of units in both participants were more strongly correlated with cursor velocity than with position (figure 3(*b*)–(*d*)). The statistical test (see section 2.6) showed that 65.3% of units were significantly correlated (*t*-test; $p < 0.01$) with both position and velocity, 5.7% with position alone and 24.7% with velocity alone. This result indicated that 95.7% of all the units detected by the microelectrode array in human motor cortex were significantly correlated with at least one of the TC kinematic parameters. The CC values for velocity were overall larger than those for position ($p << 0.01$; Wilcoxon rank sum test) as shown in figure 3(*b*). In 16 of 17 recording sessions, more units were significantly correlated with cursor velocity than position (figure 3(*c*)). Furthermore, among all 1226 units, 69.9% of units exhibited stronger correlation with velocity while only 25.9% of units showed stronger correlation with position (figure 3(*d*)). This stronger correlation with velocity was observed regardless of the training task: the random pursuit-tracking task for S3-{40, 48, 54} and A1-{85, 91} or the center-out-back task for the remainder (see table 1). These findings suggest that, for

these neuronal populations, a computer cursor may be better controlled by a person with tetraplegia if using a decoding method based on cursor velocity rather than cursor position.

3.2. Effect of training procedures on cursor control

We evaluated how cursor control was achieved through the training procedures presented in this study for different decoding algorithms. Two decoding algorithms were considered here including the position-based linear filter trained with the random pursuit-tracking task and the velocity-based Kalman filter trained with the center-out-back task. In particular, we studied how much OL training alone enabled cursor control for each decoding algorithm and how much CL training improved that performance. To quantify this, we computed the CC between the TC and the feedback cursor (FC) kinematic parameter (position or velocity depending on the algorithm used). We obtained the average CC (averaged over the horizontal and vertical axes at the optimal lag for each block, see section 2) for each of the first 4 CL blocks from 14 recording sessions (PLP and VKC) in both S3 and A1. The optimal lag averaged over 14 sessions at which the CC was maximized was 0.47 s (SD 0.4 s) which means that the TC led the FC by 0.47 s.

Figure 4 shows the comparison of the CC for the two decoding algorithms. In both participants, the CC obtained with the position-based linear filter registered a much lower value in the first CL block and improved considerably during the remaining blocks. On the other hand, the CC obtained with the velocity-based Kalman filter reached over 0.5 in the first CL block and improved only slightly afterward. Note that, in addition to having higher CCs, the velocity-based decoding results in S3 were all obtained with fewer units than the position-based results (see table 1). In our experiments, an average of 2.4 min of data (2 OL blocks) were required to train the velocity-based Kalman filter to achieve the CC > 0.5 and 8 min of data (4 OL + 4 CL blocks) were required to train the position-based linear filter to achieve similar accuracy.

This result suggests that training the velocity-based Kalman filter with the (four-target) center-out-back task required less training time compared to training the position-based linear filter with the random pursuit-tracking task. We did not investigate whether such rapid training was made possible due to decoding velocity, which was more correlated with neural activity (section 3.1), or using a relatively simpler center-out task, or both. For practical NIS development however, we can conclude that less training time is required using the velocity-based Kalman filter with the center-out-back task.

3.3. Evaluation of changes in directional tuning from training to testing

Previous animal studies have sometimes shown significant changes in directional tuning of cells between open-loop and closed-loop neural control [4,5]. Consequently, we sought to establish whether the directional tuning of cells was stable or changed between training and testing in tetraplegic humans using an NIS over a series of training and testing periods (total 30–60 min). Since here we focused on directional tuning change we analyze data from VKC sessions that explicitly decoded velocity (207 units in S3 from six sessions: S3-{254, 261, 280, 285, 286 and 288} and 257 units in A1 from three sessions: A1-{197, 216 and 224}). A statistical analysis of tuning change revealed one class of units that was consistent and another class that changed over time.

Figure 5 illustrates three example units that were consistently tuned to the TC direction across training and testing (recorded from S3-261). From the last two CL training blocks of S3-261, we sampled the spike trains of three units within a 1 s time window from the onset of the target appearance in one of four directions (0°, 90°, 180°, 270°). Nine spike trains per unit, per direction are shown in the left column of figure 5. The arrows in the circle represent

the estimated preferred direction (PD) of each unit. The right column of figure 5 shows the directional tuning of these units during testing where the four-target center-out-back task was performed. Since the NC could move to any point on the 2D task space (i.e. computer screen), the intended direction was continuously distributed from 0° to 360°. Therefore, we divided the space of the intended direction into eight equally spaced angular bins centered at (0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°). Nine 1 s spike trains for each angular bin are presented here. Visual comparison of the spike raster plots between the left and the right columns of figure 5 suggests that directional tuning of these units did not change substantially from training to testing. This also illustrates that even when trained using only four directional movements, spiking activities of these units were broadly tuned over a wider range of directions, thus enabling the NC to move in any direction after training. In contrast, figure 6 illustrates two examples of the units (from A1-197) that changed their directional tuning between the training (left) and testing (right) periods. These units showed statistically significant changes in the PD from training to testing, although the tuning depth of these units in A1 (figure 6) was not as large as those in S3 (figure 5). We remark here that these results do not conclusively demonstrate that human motor cortical neurons exhibit rapid PD changes. Rather, figures 5 and 6 show examples of the neuronal units that sustained or changed their preferred directions within a 2 h recording period.

We performed a statistical analysis of the tuning change for all 464 units (207 units in S3 and 257 in A1) to find how many units changed or sustained directional tuning and how much their preferred directions and tuning depths were changed. First, the percentage of units that were significantly tuned to direction (F-test, $p < 0.01$) increased from 69.6% (144 units) during training to 80.2% (166 units) during testing in S3 and from 35.0% (90 units) to 63.4% (163 units) in A1. Of those units that had been tuned during training (144 units in S3 and 90 in A1), 4.9% (7 units) in S3 and 24.4% (22 units) in A1 lost directional tuning during testing. Of those units that were *not* tuned during training (63 units in S3 and 167 units in A1), 46.0% (29 units) in S3 and 56.9% in A1 (95 units) gained directional tuning during testing. Of those units that were tuned during training, 95.1% (137 units) in S3 and 75.6% in A1 (68 units) remained directionally tuned from training to testing.

Second, among those units that remained directionally tuned from training to testing, we evaluated how much their PD changed. Note that the PD was deemed to be significantly changed if the two 95% confidence intervals computed in training and testing respectively did not overlap (see section 2.8). We found that 72.9% (100 out of 137 units) in S3 and 42.7% (29 out of 68 units) in A1 showed no significant change in PD (95% confidence). For the other units in which the PD significantly changed, the average change in angle was 34.1° (SD 20.4°) for S3 and 99.7° (SD 56.6°) for A1, respectively. Figure 7 summarizes these results.

Finally, we evaluated the change of the tuning depth of those units that remained directionally tuned (137 units in S3 and 68 in A1) between training and testing. We found that 24.8% (34 units) in S3 and 33.8% (23 units) in A1 changed their tuning depths significantly (95% confidence). Among these units, 82.4% (28 units) in S3 and 82.6% (19 units) in A1 increased the tuning depth and 17.6% (6 units) in S3 and 17.4% (4 units) in A1 decreased the tuning depth. On average, the tuning depth increased by 61.9% (SD 68.2%) from training to testing (over 34 units) in S3 and by 130.8% (SD 122.1%, over 23 units) in A1, respectively; this increase in the tuning depth was statistically significant (paired t-test, $p \approx 7.8 \times 10^{-6}$ for S3 and $p \approx 3.8 \times 10^{-5}$ for A1). The increases in the tuning depth and in the number of tuned units during testing suggests that directional tuning in MI firing activity became stronger between training, where the participants imagined following the visually guided cursor movements, to testing, where they volitionally moved the cursor.

Although we did not perform an extended study of the relationship between consistency in tuning and cursor control performance, we posit that the larger number of tuning changes in the neuronal population of A1 might partially explain A1's poorer cursor control performance relative to S3 (see the following section).

3.4. The velocity-based Kalman filter versus the position-based linear filter

Based on the tuning analysis above we compared on-line cursor control in tetraplegic humans using the velocity-based Kalman filter with the position-based linear filter adopted from Hochberg *et al* [13]. We found that cursor control performance was significantly improved with the velocity-based Kalman filter for both participants. We evaluated performance in the four-target acquisition task across those recording sessions in which the identical task configuration was used (nine sessions for S3, five for A1). The number of target acquisition assessment trials was 80 per session except S3-285 (44 trials), S3-287 (73), A1-197 (30), A1-216 (12) and A1-224 (43). In each trial, the participants had to acquire the target before a 7 s timeout.

Figures 8–11 illustrate the performance of the two decoding methods. The figures present the decoded NC paths starting from the center and moving to each of four peripheral targets for all 14 recording sessions, along with the mean NC paths. These figures show that the NC paths decoded with the velocity-based Kalman filter were much straighter and smoother than those produced by the position-based linear filter in both participants. Note that the NC movements using the velocity-based Kalman filter were achieved with fewer units: the ratios of the average unit counts for the position-based linear filter sessions compared with the later velocity-based Kalman filter sessions were 162:35 for S3 and 110:86 for A1.

Using pointing device performance measures based on the ISO 9241–9 standard as well as other cursor control measures (see section 2), we quantified the performance of closed-loop neural cursor control for each participant, as summarized in table 2. The *control* chance rate was 9.8% for the position-based linear filter, which was on a par with previously reported results in a different participant [13]. For the velocity-based Kalman filter, the control rate was 3.9%. This lower control rate suggests that the better controlled NC trajectories produced by the velocity-based Kalman filter reduced the chance of selecting an incorrect target; this is a significant advantage in a real task where many targets are simultaneously placed and false targets could be selected (see section 2 for the estimation of the control rate).

For S3, target acquisition accuracy improved using the velocity-based Kalman filter over using the position-based linear filter. S3 failed to acquire 52 out of 240 targets before the timeout (ER = 21.7%) using the position-based linear filter over three sessions versus 60 out of 436 targets (ER = 13.8%) using the velocity-based Kalman filter over six sessions, reducing the absolute value of ER by 7.9%. For the observed target acquisition rate (78.3%) using the position-based linear filter, the baseline chance rate for the four-target center-out task was 19.6% (i.e. a quarter of the target acquisition rate). However, the adjusted chance rate obtained from simulation was 25.9%, higher than the baseline chance rate by 6.3%. This implies that there was a 6.3% chance that an incorrect target was selected by accident due to unsteady NC behavior. The baseline chance rate for the velocity-based Kalman filter (target acquisition rate = 86.2%) was 21.6%. The adjusted chance rate was 23.9%, 2.3% higher than the baseline chance rate. So, the chance of selecting an incorrect target was reduced with the velocity-based Kalman filter (see section 2 for the estimation procedure of the chance rate). The significant target acquisition rate (STAR), which was obtained by subtracting the control rate from the target acquisition rate, was 68.5% (= 78.3–9.8) for the position-based linear filter versus 82.3%(=86.2–3.9) for the velocity-based Kalman filter. Cursor movement was slightly slower when the velocity-based Kalman filter was used; overall mean

movement time (MT) increased by 0.49 s (one-tailed *t*-test; $p < 10^{-4}$). Note that MT was only assessed (and only defined) for successful target acquisition trials. Analysis of the NC movement paths from the starting point to the target demonstrated that NC movement decoded by the velocity-based Kalman filter produced smoother trajectories compared to that decoded by the position-based linear filter. All four measurements of the cursor path accuracy, including orthogonal direction change (ODC), movement direction change (MDC), movement error (ME) and movement variability (MV), were significantly smaller using the velocity-based Kalman filter (one-tailed *t*-test; $p < 0.01$).

A1 missed 43 out of 160 targets (ER = 26.3%) using the position-based linear filter versus 27 out of 85 targets (ER = 31.8%) using the velocity-based Kalman filter, increasing overall ER by 4.9%. However, the chance rate was 7.6% higher than the baseline chance rate (18.3%) with the position-based linear filter and 6.9% higher than the baseline chance rate (17.1%) with the velocity-based Kalman filter. This suggests that the effect of unstable NC control on target acquisition performance was similar for the two decoders for A1. The STAR was 63.9% (= 73.7 − 9.8) for the position-based linear filter versus 64.3% (= 68.2 − 3.9) for the velocity-based Kalman filter, again demonstrating a similar target acquisition performance achieved by volitional neural control. MT was reduced by 0.03 s using the velocity-based Kalman filter but this difference was not statistically significant (one-tailed *t*-test; $p = 0.45$). All four fine measures characterizing the NC paths demonstrated that the NC movements decoded by the velocity-based Kalman filter were smoother (significantly smaller ODC, MDC, ME and MV; one-tailed *t*-test; $p < 0.01$) than using the position-based linear filter in participant A1.

The longer MT with the velocity-based decoder could be attributed to two factors: slower cursor speed or difficulty in holding the cursor still to select the target. To test these two hypotheses, we compared the decoded speed (i.e. the length of the 2D decoded velocity vector) for the position-based linear filter and the velocity-based Kalman filter. A comparison of the distributions of the decoded speed revealed that the position-based linear filter produced significantly (Wilcox rank sum, $p << 0.1$) faster cursor speeds than the velocity-based Kalman filter; the median speed was 64.9 pixel s^{-1} for linear position and 39.0 pixel s^{-1} for Kalman velocity, respectively (figure 12(*a*)). We further examined whether the decoded cursor speed was related to the spatial position of the cursor on the screen. To do so we divided the screen space into small regions and computed the mean speed within each region. We found that the cursor speed of the position-based linear filter was highest near the boundaries of the screen (figure 12(*b*)) while the cursor speed of the velocity-based Kalman filter showed the highest speed between the center and the four targets and a lower speed around the targets (figure 12(*c*)). The latter is what we would expect for a target acquisition task. This comparison suggests that although cursor speed decoded with the velocity-based Kalman filter was relatively slow, it tended to be more accurately controlled in terms of performing the task.

3.5. Contribution of kinematic parameters versus the decoding algorithm

The performance evaluation presented above demonstrated that cursor control was improved using the velocity-based Kalman filter relative to the position-based linear filter. However, these results did not differentiate between the effects of the decoding algorithm (linear versus Kalman filter) and the kinematic parameter (position versus velocity). Therefore, we conducted three ‘VLKC’ recording sessions with S3 (S3-408, S3-412 and S3-418) in which velocity-based linear and velocity-based Kalman filters were evaluated sequentially to specifically provide a comparison of the decoding algorithms only. In each session, either the Kalman or the linear filter was first trained and tested with the eight-target acquisition task. Then, the other filter was again trained and tested using exactly the same training (center-out-back) and testing tasks. Each filter had its own training blocks. During training

we identified the neuronal units with firing rates > 1 Hz during the first two open-loop blocks and used them to build and test the decoding algorithms. Hence, the number of units used for each filter could be slightly different even during the same day (see table 3). We changed the order of the decoding algorithm for each day: Kalman followed by linear filters for S3-408; and linear followed by Kalman filters for S3-412 and S3-418. Note that we were unable to test both decoding algorithms with position as the kinematic variable; S3 was unable to gain control of a cursor using position decoding in later trial sessions where an average of 35 units were recorded.

Individual NC paths along with the mean path to each target are shown in figure 13. The performance evaluation of each velocity-based filter is described in table 3. Over three sessions, the performance using the velocity-based linear filter exhibited slower (higher MT; *t*-test, $p < 0.01$) and less accurate (the absolute value of ER increased by 32.9%) NC control compared to the velocity-based Kalman filter. A false target was never selected using the velocity-based linear filter (90 trials) while false targets were selected three times out of 102 trials (2.9%) using the velocity-based Kalman filter. The STAR derived from the control rate was 51.1% for the linear filter and 71.1% for the Kalman filter, respectively. These results suggest that the Kalman filter enabled more rapid cursor control than the linear filter when both filters decoded cursor velocity, which led to better target acquisition performance for the Kalman filter within a fixed time limit.

The NC showed a larger movement direction change (MDC) with the linear filter than with the Kalman filter (*t*-test, $p < 0.01$) reflecting the 'shaky' NC trajectories observed with the velocity-based linear filter (see figure 12). However, the slow NC movements of the linear filter did not register more orthogonal direction changes (ODC) than those by the Kalman filter (*t*-test, $p > 0.01$), which indicates the NC decoded by the linear filter, although slower, moved forward in the correct direction. Movement variability (MV) and movement error (ME) using the velocity-based linear filter were slightly (<5 mm) smaller than those using the velocity-based Kalman filter (*t*-test, $p < 0.01$). These results indicate that the poorer target acquisition performance by the velocity-based linear filter was due to the fact that the NC moved too slowly to reach the target in a given time limit. Note that we trained both the linear and Kalman filters using cursor velocity and the parameters of these models were both optimal for decoding this velocity. Consequently, we did not apply any *post-hoc* gain to scale the decoded velocities. Such an approach might be used to increase cursor speed (for both decoders) and may be worth considering in clinical applications. Similarly, improved methods for decoding speed might be adopted.

In this study, the linear filter was tested with both position and velocity kinematics. However, experiments examining the velocity-based linear filter employed a more difficult task than those for the position-based linear filter; smaller targets were located at a larger distance from the center, which contributed to longer MT and, consequently, higher ER (due to timeout) for reaching and dwelling (see the performance of the velocity-based Kalman filter for different task setups in tables 2 and 3). Hence, it is nontrivial to directly compare performance between the position-based and velocity-based linear filters. However, we observed that even with the increased task difficulty and longer movement times, the velocity-based linear filter exhibited better ODC, MDC, ME and MV than the position-based linear filter; these measures are relatively independent of the target size and distance. Therefore, with respect to these measures, the performance difference between using the velocity-based Kalman and the velocity-based linear filters was not as large as that between the velocity-based and position-based linear filters. This observation suggests that decoding appropriate kinematic parameters—i.e. velocity versus position—may play a more important role than the decoding algorithm itself in improving cursor control in an NIS.

4. Discussion

We found that intracortical control of a computer cursor from MI in two humans with tetraplegia was improved when decoding cursor velocity with a Kalman filter rather than when decoding position with a linear filter. The Kalman velocity method provided smoother and more accurate cursor control than decoding position using the linear filter in two participants (one with brainstem stroke and the other with ALS) during a four-target center-out task. The study (in S3) of the effect of the decoding algorithm on cursor control revealed that cursor speed was higher with the Kalman filter, which consequently led to a higher target acquisition rate within a 10 s timeout period compared to using the linear filter decoding velocity. The NC paths estimated by the Kalman filter were also less shaky than those produced by the linear filter, resulting in lower MDC measures. However, other fine measures of the NC path related to the variability and deviation from a straight line showed similar performance levels between the two algorithms. Moreover, both methods exhibited superior performance to the case when position was decoded using the linear filter even though the position decoding sessions used many more units.

Hence, from these results in two participants, we found that selection of kinematic parameters had a more profound impact on cursor control performance than the choice of the decoding algorithm. We further found that the participants achieved reasonable cursor control (after visually-guided open-loop training) more rapidly with the velocity-based Kalman filter decoding algorithm thus requiring less training time than the position-based linear filter algorithm.

Our approach to decoding velocity (or, equivalently, direction and speed) from human motor cortex as a cursor control signal was based on many previous non-human primate studies of neuronal modulation. Those studies found modulation in MI neuronal firing activity to be related to movement direction during one-dimensional arm movements [39,40], multi-dimensional point-to-point reaching movements [16–19,41] and drawing and pursuit-tracking movements [14,42]. In particular, the work of Taylor *et al* [5] and following work [8,43] on real-time closed-loop cortical control of a prosthetic device (e.g. a robotic arm) by neurologically intact non-human primates established the feasibility of using velocity decoding from neural population activity to control prosthetic devices. Our finding that decoding velocity provided better cursor control than decoding position supports these previous findings and extends them to humans. Beyond previous work, the present study also shows that cursor velocity was better extracted from the motor cortical neurons during purely imagined (or attempted) movements in tetraplegic humans. This was evaluated with an explicit comparison between velocity and position decoding during closed-loop cursor control.

Previous studies, as well as our own here, leave open the question as to what is the ‘best’ control variable to decode MI activity. For instance, we have observed during a single recording session in A1 that adding cursor acceleration to the kinematic parameters of the Kalman filter also resulted in superior performance when compared with position-based decoding (unpublished observation). In addition, when cursor position and velocity were decoded simultaneously by the Kalman filter, the performance was inferior to the case when only velocity was decoded (observed in multiple recording sessions in S3 and A1, with 67–90 units recorded). This suggests not only that choosing an appropriate kinematic representation of cursor movement may be important to achieve smooth cursor control (e.g. velocity versus position), but also that exploring and modeling additional relationships between neural signals and cursor kinematics may lead to further improvements in cursor control. At this point, we conclude that an NIS that is informed by an analysis of neural

population coding can result in improved cursor control. This approach should extend to other types of prosthetic devices such as robot arms as well.

As in the first work by Hochberg *et al* [13], selection of targets was performed by briefly holding the NC on the target ('dwell'). Such an interface has serious limitations in practice and most commercial pointing devices have some specific mechanism for target selection ('clicking'). Here, we found that the chance rate for target selection with dwell could be high when there was only one selectable target on the screen. The addition of multiple targets (as exist in most commercial software interfaces, such as common word processing software) increased the chance of a false selection making the need for a specific click function more important. Implementing a discrete neural 'click' is a topic of our current research; initial progress has been reported in Kim *et al* [44].

In an earlier study of similar data from humans with tetraplegia, Truccolo *et al* [26] found no significant difference between position and velocity tuning of MI neurons. It is important to note that the Truccolo *et al* analysis was performed on closed-loop (CL) blocks in which the subject viewed a decoded feedback cursor (FC) that was controlled by the position-based linear filter. In contrast, during OL blocks, with no FC present, we found significantly more velocity tuned units. It is possible that the position-based decoding model in the Truccolo *et al* study biased the population toward position tuning thus reducing tuning to velocity. This suggests an interesting direction for future study: does the use of a particular kinematic representation in the decoding method change the tuning properties of MI neurons?

The design and evaluation of decoder training paradigms for a human NIS require more extensive study. For the sessions presented in this paper, two training tasks were used depending upon which kinematic parameter was being decoded. The center-out-back task, even with more than four targets, may not be appropriate for training a position-based decoder since the position samples of the TC only cover a limited space of the screen. Rather, the random pursuit-tracking task provides a broader range of the TC position data for position-based decoder training. Indeed, our preliminary on-line results indicated that a position-based decoder trained with the random pursuit-tracking task yielded better cursor control than a position-based decoder trained with the center-out-back task. On the other hand, performing the random pursuit (or step) tracking task may not dissociate velocity from position as effectively as the center-out-back task during a finite training period (up to ~10 min). We have not extensively studied the use of the random pursuit-tracking task to train a velocity-based decoder, but we anticipate longer training times. A key finding here was that the velocity-based decoder required very little training, as the first few OL training blocks (~2.4 min) produced usable cursor control. The question remains as to whether improved feedback methods or other filter (or participant) training protocols could further improve the accuracy of control, even without further improvements in decoding algorithms.

It is important to note that we did not present how to train the human user of an NIS in this study. It is unknown how a decoder training process affects the cognitive aspects of the human user. Although we speculate that there must be some types of 'training' in the participants to achieve cursor control within or across the trial days, we have not systematically quantified them. However, we foresee that human factors in training will become more important as we advance the NIS to enable the control of more complex effectors (e.g. robotic arms). Our future study will investigate this.

In terms of decoding algorithms, both the linear filter and Kalman filter are simple (linear) methods that may not be optimal for neural cursor control. In off-line analysis of non-human primate recordings, several authors have observed that more complex decoding methods can improve on these two standard techniques [45–49]. The improvement, however, may come

with additional computational cost or the need for more training data and, consequently, longer training times. Improved decoding algorithms that maintain the benefits of the Kalman filter are a topic of our current research.

The variation of neural spiking activity (e.g. changes in tuning between training and testing; see section 3.3) within a single day or across days suggests that the sources of such variation should be studied further. Some possible sources of variability include: (1) some of the recorded neural waveforms were of low amplitude and consequently may have been difficult to sort into single units. For multi-units, changes in neural modulations could be due to changes in the relative rates of multiple cells rather than intrinsic changes to any given cell. (2) With multi-unit recordings, if waveforms from one cell are close to the threshold for acceptance, small fluctuations in the waveform could change the overall multi-unit activity on a given electrode. (3) There may be task-related variation in neural activity between training and testing. The change from viewing the TC to active closed-loop control of the NC might change the modulation of neural firing rates. This could exacerbate the issues related to multi-unit recordings mentioned above. (4) The visual feedback between training and testing also changed. In the testing sessions, only the NC was visible while in training both the FC and TC appeared at the same time. While the subject was instructed to imagine moving the TC during training, the presence of the FC might have altered their intended movement. (5) There may be intrinsic non-stationarity in the neural population that could also change the directional tuning. (6) There may be unmeasured and un-modeled electrical or mechanical factors that vary over time and change the nature of the recorded signals. For example, any changes in the array placement relative to the neural tissues could change the units recorded from one time to another. Based on our observations this appears very unlikely.

It is commonly anticipated that, all else being equal, decoder performance can be improved with the addition of more spiking units [3,4,7,25]. The position-based linear filter decoding results reported for S3 were derived from recording sessions approximately 2 months after implantation, during which time the number of identifiable units or multi-units ranged from 147–179. On the other hand, the velocity-based Kalman filter decoding results were reported for the sessions conducted 10–11 months after implantation during which time the number of units ranged from 13–80. During some recording sessions in this latter period, which were not presented in this report, we found that position-based decoding yielded completely inaccurate cursor movements so that no control of a cursor was achieved (position was decoded using both the linear filter or Kalman filter method). Consequently, in our analysis we compared position-based decoding with relatively high unit counts to velocity-based decoding with relatively low unit counts. Despite this difference in unit counts we found that velocity-based decoding was superior to position-based decoding. Thus one primary advantage of velocity-based cursor control, regardless of the decoding method, was its ability to provide usable control with relatively small numbers of units. With comparable unit counts we expect this difference in performance between position and velocity decoding to be even larger.

Acknowledgments

This work is partially supported by NIH-NINDS R01 NS 50867-01 as part of the NSF/NIH Collaborative Research in Computational Neuroscience Program, NIHNINDS NS25074, the Katie Samson Foundation, and by the Office of Naval Research (awards N0014-04-1–082 and N0014-07-1–0803). This material is based upon work supported in part by the Office of Research and Development, Rehabilitation R&D Service, Department of Veterans Affairs (JDS, LRH). We also thank the European Neurobotics Program FP6-IST-001917 and Rehabilitation Research and Development Service, Department of Veterans Affairs. The clinical trial from which these data are derived is sponsored by Cyberkinetics Neurotechnology Systems, Inc. We thank Gregory Shakhnarovich, Frank Wood and Wilson Truccolo for comments and assistance. We also thank Abe Caplan, Eric Peterson, Michael Fritz and Cyberkinetics Neurotechnology Systems, Inc. for running the recording sessions described here.

References

1. Serruya MD, Hotsopoulos NG, Paninski L, Fellows MR, Donoghue JP. Instant neural control of a movement signal. Nature 2002;416:141–2. [PubMed: 11894084]
2. Chapin JK, Moxon KA, Markowitz RS, Nicolelis MAL. Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex. Nature Neurosci 1999;2:664–70. [PubMed: 10404201]
3. Wessberg J, Stambaugh CR, Kralik JD, Beck PD, Laubach M, Chapin JK, Kim J, Biggs SJ, Srinivasan MA, Nicolelis MAL. Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. Nature 2000;408:361–5. [PubMed: 11099043]
4. Carmena JM, Lebedev MA, Crist RE, O'Doherty JE, Santucci DM, Dimitrov DF, Patil PG, Henriquez CS, Nicolelis MAL. Learning to control a brain–machine interface for reaching and grasping by primates. PLoS Biol 2003;1:E42. [PubMed: 14624244]
5. Taylor DM, Helms Tillery SI, Schwartz AB. Science 2002;296:1829–32. [PubMed: 12052948]
6. Santhanam G, Ryu SI, Yu BM, Afshar A, Shenoy KV. A high-performance brain–computer interface. Nature 2006;442:195–8. [PubMed: 16838020]
7. Musallam S, Corneil BD, Greger B, Scherberger H, Andersen RA. Cognitive control signals for neural prosthetics. Science 2004;305:258–62. [PubMed: 15247483]
8. Velliste M, Perel S, Spalding MC, Whitfort AS, Schwartz AB. Cortical control of a prosthetic arm for self-feeding. Nature 2008;453:1098–101. [PubMed: 18509337]
9. Donoghue JP. Connecting cortex to machines: recent advances in brain interfaces. Nature Neurosci. Suppl 2002;5:1085–8.
10. Schwartz AB, Cui XT, Weber DJ, Moran DW. Brain-controlled interfaces: movement restoration with neural prosthetics. Neuron 2006;52:205–20. [PubMed: 17015237]
11. Chapin JK. Using multi-neuron population recordings for neural prosthetics. Natl Neurosci 2004;7:452–5.
12. Nicolelis MAL. Actions from thoughts. Nature 2001;409:403–7. [PubMed: 11201755]
13. Hochberg LR, Serruya MD, Friehs GM, Mukand JA, Saleh M, Caplan AH, Branner A, Chen D, Penn RD, Donoghue JP. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature 2006;442:164–71. [PubMed: 16838014]
14. Paninski L, Fellows MR, Hotsopoulos NG, Donoghue JP. Spatiotemporal tuning of motor cortical neurons for hand position and velocity. J. Neurophysiol 2004;91:515–32. [PubMed: 13679402]
15. Kettner RE, Schwartz AB, Georgopoulos AP. Primate motor cortex and free arm movements to visual targets in three-dimensional space: III. Positional gradients and population coding of movement direction from various movement origins. J. Neurosci 1988;8:2938–47. [PubMed: 3411363]
16. Humphrey DR, Schmidt DM, Thompson WD. Predicting measures of motor performance from multiple cortical spike trains. Science 1970;170:758–62. [PubMed: 4991377]
17. Georgopoulos AP, Kalaska JF, Caminiti R, Massey JT. On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. J. Neurosci 1982;2:1527–37. [PubMed: 7143039]
18. Ashe J, Georgopoulos AP. Movement parameters and neural activity in motor cortex and area 5 Cereb. Cortex 1994;4:590–600.
19. Moran DW, Schwartz AB. Motor cortical representation of speed and direction during reaching. J. Neurophysiol 1999;82:2676–92. [PubMed: 10561437]
20. Fu QG, Flament D, Coltz JD, Ebner TJ. Temporal encoding of movement kinematics in the discharge of primate primary motor and premotor neurons. J. Neurophysiol 1995;73:836–54. [PubMed: 7760138]
21. Sergio LE, Kalaska JF. Changes in the temporal pattern of primary motor cortex activity in a directional isometric force versus limb movement task. J. Neurophysiol 1998;80:1577–83. [PubMed: 9744964]
22. Todorov E. Direct cortical control of muscle activation in voluntary arm movements: a model. Nat. Neurosci 2000;3:391–8. [PubMed: 10725930]

23. Lebedev MA, Carmena JM, O'Doherty JE, Zacksenhouse M, Henriques CS, Principe JC, Nicolelis MAL. Cortical ensemble adaptation to represent velocity of an artificial actuator controlled by a brain–machine interface. J. Neurosci 2005;25:4681–93. [PubMed: 15888644]
24. Wu W, Shaikhouni A, Donoghue JP, Black MJ. Closed loop neural control of cursor motion using a Kalman filter. Proc. Conf. IEEE Eng. Med. Biol. Soc 2004:4126–9.
25. Wu W, Gao Y, Bienenstock E, Donoghue JP, Black MJ. Bayesian population decoding of motor cortical activity using a Kalman filter. Neural. Comput 2006;18:80–118. [PubMed: 16354382]
26. Truccolo W, Friehs GM, Donoghue JP, Hochberg LR. Primary motor cortex tuning to intended movement kinematics in humans with tetraplegia. J. Neurosci 2008;28:1163–78. [PubMed: 18234894]
27. Douglas, SA.; Kirkpatrick, AE.; MacKenzie, SI. Testing pointing device performance and user assessment with the ISO 9241, Part 9 standard. Proc. SIGCHI Conf. on Human Factors in Computing Systems (CHI'99, Pittsburgh, PA, United States); 1999. p. 215-22.
28. Mackenzie, IS.; Kauppinen, T.; Silfverberg, M. Accuracy measures for evaluating computer pointing devices. Proc. the SIGCHI Conf. on Human Factors in Computing Systems (CHI'01, Seattle, WA, United States); 2001. p. 9-16.
29. Warland DK, Reinagel P, Meister M. Decoding visual information from a population of retinal ganglion cells. J. Neurophysiol 1997;78:2336–50. [PubMed: 9356386]
30. Gelb, A. Applied Optimal Estimation. MIT Press; Cambridge, MA: 1974.
31. Suner S, Fellows MR, Vargas-Irwin C, Nakata K, Donoghue JP. Reliability of signals from chronically implanted, silicon-based electrode array in non-human primate primary motor cortex. IEEE Trans. Neural. Syst. Rehabil. Eng 2005;13:524–41. [PubMed: 16425835]
32. Krakauer JW, Ghilardi MF, Ghez C. Independent learning of internal models for kinematic and dynamic control of reaching. Nature Neurosci 1999;2:1026–30. [PubMed: 10526344]
33. Tong C, Flanagan JR. Task-specific internal models for kinematic transformations. J. Neurophysiol 2002;90:578–85. [PubMed: 12904486]
34. Wolpaw JR, McFarland DJ. Control of a two-dimensional movement signal by a noninvasive brain–computer interface in humans. PNAS 2004;101:17849–54. [PubMed: 15585584]
35. Kalman RE. A new approach to linear filtering and prediction problems. J. Basic Eng 1960;82:35–45.
36. Pearson K. Mathematical contributions to the theory of evolution: III. Regression, heredity and panmixia. Phil. Trans. R. Soc. Lond. A 1896;187:253–318.
37. Hotelling H. New light on the correlation coefficient and its transforms. J. R. Stat. Soc. B 1953;15:193–232.
38. Efron B, Tibshirani R. Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. Stat. Sci 1986;1:54–75.
39. Evarts EV, Fromm C, Kroller J, Jennings J. Relation of pyramidal tract activity to force exerted during voluntary movement. J. Neurophysiol 1968;31:14–27. [PubMed: 4966614]
40. Fetz EE, Finocchio DV, Baker MA, Soso MJ. Sensory and motor responses of precentral cortex cells during comparable passive and active joint movements. J. Neurophysiol 1980;43:1070–89. [PubMed: 6766994]
41. Schwartz AB, Kettner RE, Georgopoulos AP. Primate motor cortex and free arm movements to visual targets in three-dimensional space: I. Relations between single cell discharge and direction of movement. J. Neurosci 1988;8:2913–27. [PubMed: 3411361]
42. Moran DW, Schwartz AB. Motor cortical activity during drawing movements: Population representation during spiral tracing. J. Neurophysiol 1999;82:2693–704. [PubMed: 10561438]
43. Taylor DM, Helms Tillery SI, Schwartz AB. Information conveyed through brain-control: cursor versus robot. IEEE Trans. Neural Syst. Rehabil. Eng 2003;11:195–9. [PubMed: 12899273]
44. Kim SP, Simeral JD, Hochberg LR, Donoghue JP, Friehs GM, Black MJ. Multi-state decoding of point-and-click control signals from motor cortical activity in a human with tetraplegia. Proc. IEEE EMBS Conf. Neural Eng 2007:486–9.
45. Gao Y, Black MJ, Bienenstock E, Shoham S, Donoghue JP. Probabilistic inference of hand motion from neural activity in motor cortex. Adv. Neural Inf. Process. Syst 2002;14:221–8.

46. Brockwell AE, Rojas AL, Kass RE. Recursive Bayesian decoding of motor cortical signals by particle filtering. J. Neurophysiol 2004;91:1899–907. [PubMed: 15010499]

47. Yu BM, Kemere C, Santhanan G, Afshar A, Ryu SI, Meng TH, Sahani M, Shenoy KV. Mixture of trajectory models for neural decoding of goal-directed movements. J. Neurophysiol 2007;97:3763–80. [PubMed: 17329627]

48. Wu W, Black MJ, Mumford D, Gao Y, Bienenstock E, Donoghue JP. Modeling and decoding motor cortical activity using a switching Kalman filter. IEEE Trans. Biomed. Eng 2004;51:933–42. [PubMed: 15188861]

49. Kim SP, Sanchez JC, Rao YN, Ergodmus D, Principe JC, Carmena JM, Levedev MA, Nicolelis MAL. A comparison of optimal MIMO linear and nonlinear models for brain–machine interfaces. J. Neural. Eng 2006;3:145–161. [PubMed: 16705271]

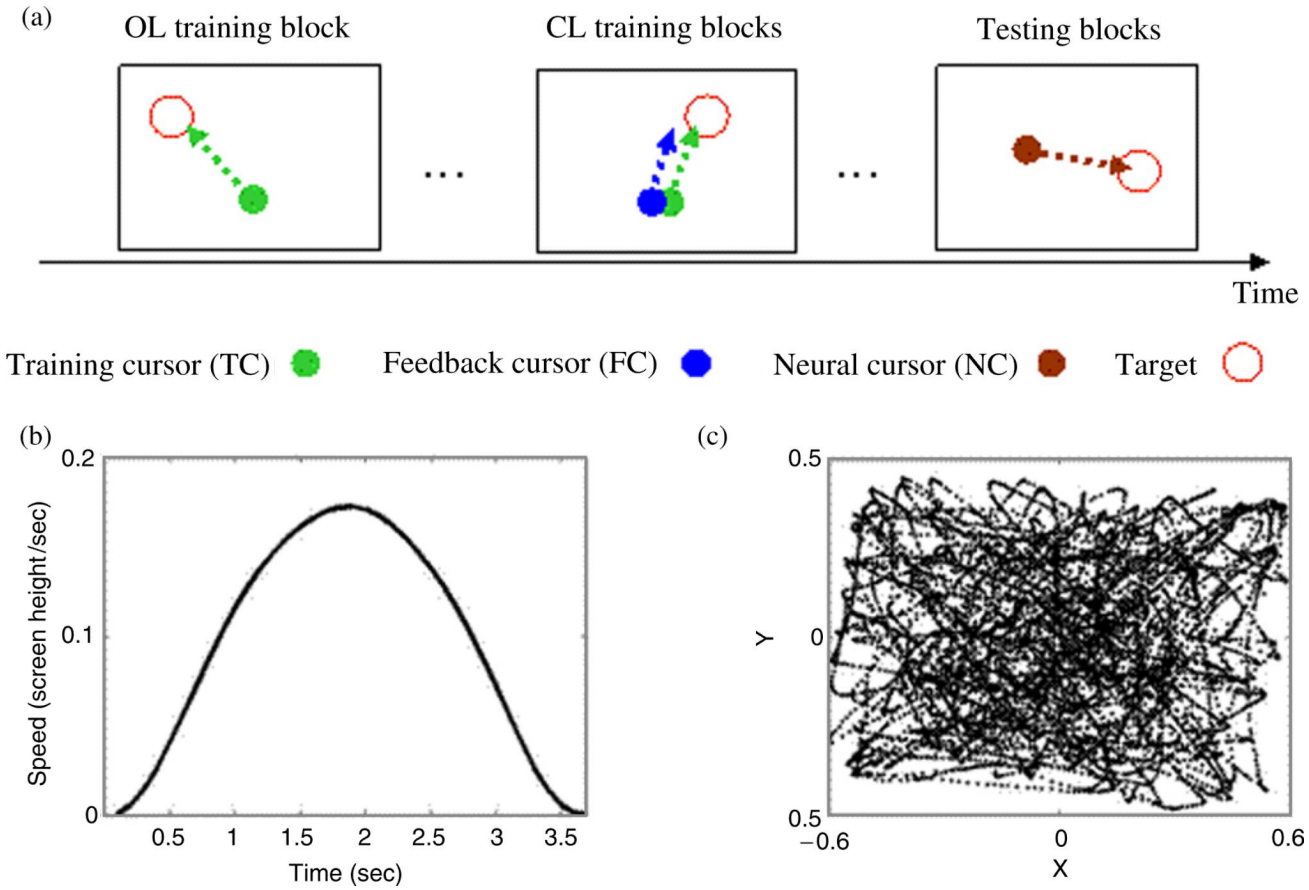

Figure 1.

Training paradigm. (*a*) Illustrations of a series of open-loop (OL) training (left), closed-loop (CL) training (middle) and testing blocks (right). In the OL block (1–1.5 min period of recording), the training cursor (TC) was moved by a technician or a computer program to reach a target. After one to four OL blocks, CL training blocks included a feedback cursor (FC), controlled by decoded neural signals in real time, that was also displayed on the monitor to provide visual feedback to the participants. In testing blocks, the participants volitionally controlled a neural cursor (NC) to reach and dwell (for 500 ms) on targets. (*b*) An example of the computer-generated speed profile for the TC. Here, a speed value of 0.1 s^{-1} represents moving 10% of the screen height per second. Across all the sessions included in this study, the speed profile duration varied from 1.5 s to 4.5 s. (*c*) An example of the TC positions collected for 8 min during the random pursuit-tracking task used for training position-based decoding algorithm (from the session S3-40). The screen space was normalized to [−0.6, 0.6] in the horizontal axis (denoted X here) and [−0.5, 0.5] in the vertical axis (*Y*).

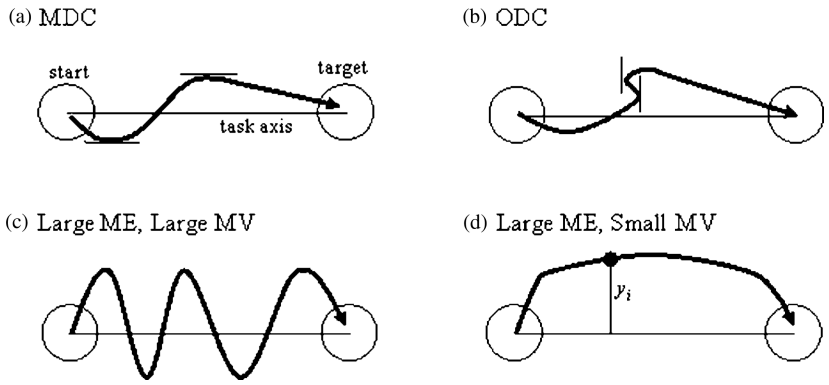

Figure 2.

Illustration of the NC path performance measures. (*a*) MDC counts the number of movement direction changes parallel to the task axis (for instance, MDC = 2 in this example). (*b*) ODC counts the number of direction changes orthogonal to the task axis (ODC = 2). (*c*)–(*d*) ME measures movement deviation by computing the average of the absolute deviations $|y_i|$ from the task axis, and MV measures the standard deviation of y_i . Two cases are illustrated in which both ME and MV are large (*c*) or ME is large but MV is small (*d*). (Illustration large derived from MacKenzie *et al* [28].)

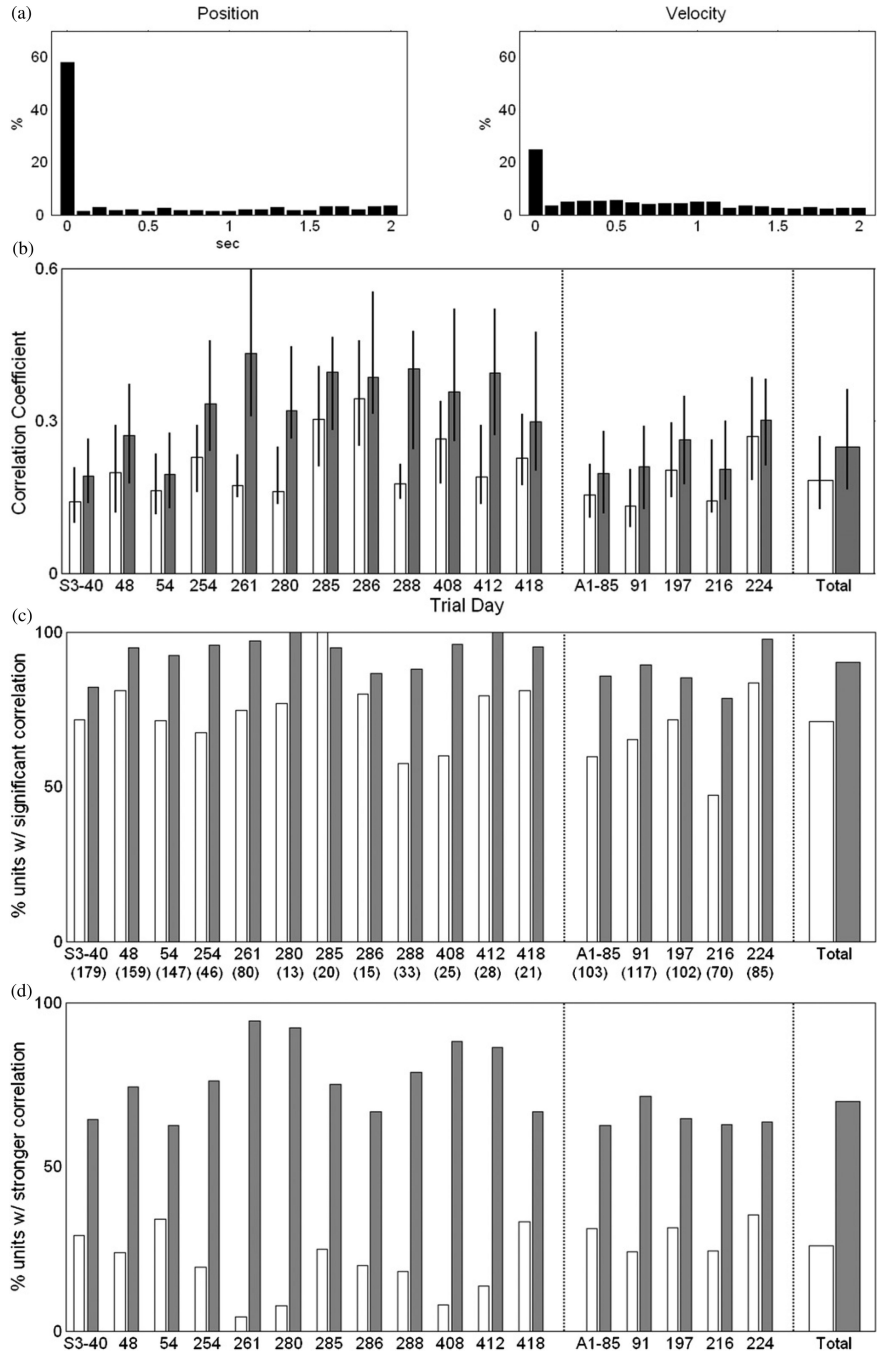

Figure 3.

Analysis of kinematic tuning during training. (*a*) Histograms showing the percentage of units with optimal lags ranging from 0 to 2 s (1126 neuronal units from 17 recording sessions in two participants (S3 and A1)). (*b*) The correlation coefficients (CC) between individual firing rates and kinematic parameters—position (white) versus velocity (gray)—during open-loop training blocks. The median of the CC values is represented by bars and 25% and 75% percentiles by vertical lines. (*c*) The percentage of units significantly correlated (methods) with either position or velocity ($p < 0.01$; *t*-test). The number of units (*N*) recorded in each session is denoted in parentheses. (*d*) The percentage of units that

showed stronger (and significant) correlation with one of position or velocity compared to the other.

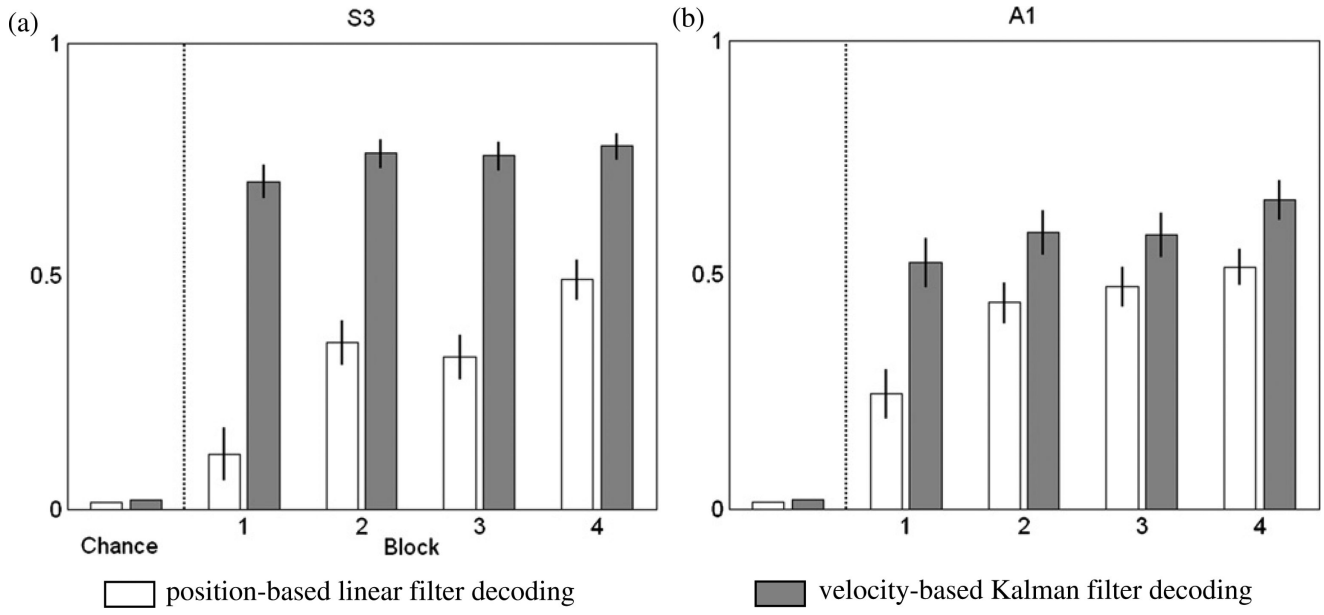

Figure 4.

Decoding performance improvement with training. Evolution of the average CC between the TC and the FC trajectories of position or velocity during closed-loop (CL) training blocks for S3 (*a*)and A1(*b*). CC = 1 represents perfect correlation between the TC and the FC. The CC was measured for the two groups of CL blocks separated based on the decoding method used: the position-based linear filter (white) and the velocity-based Kalman filter (gray). The bars to the left side of a vertical dashed line illustrate the chance level for decoding accuracy (methods) for each decoding method. The bars and the vertical lines to the right side represent the average CC and the 95% confidence interval during each of the 1st to the 4th CL blocks.

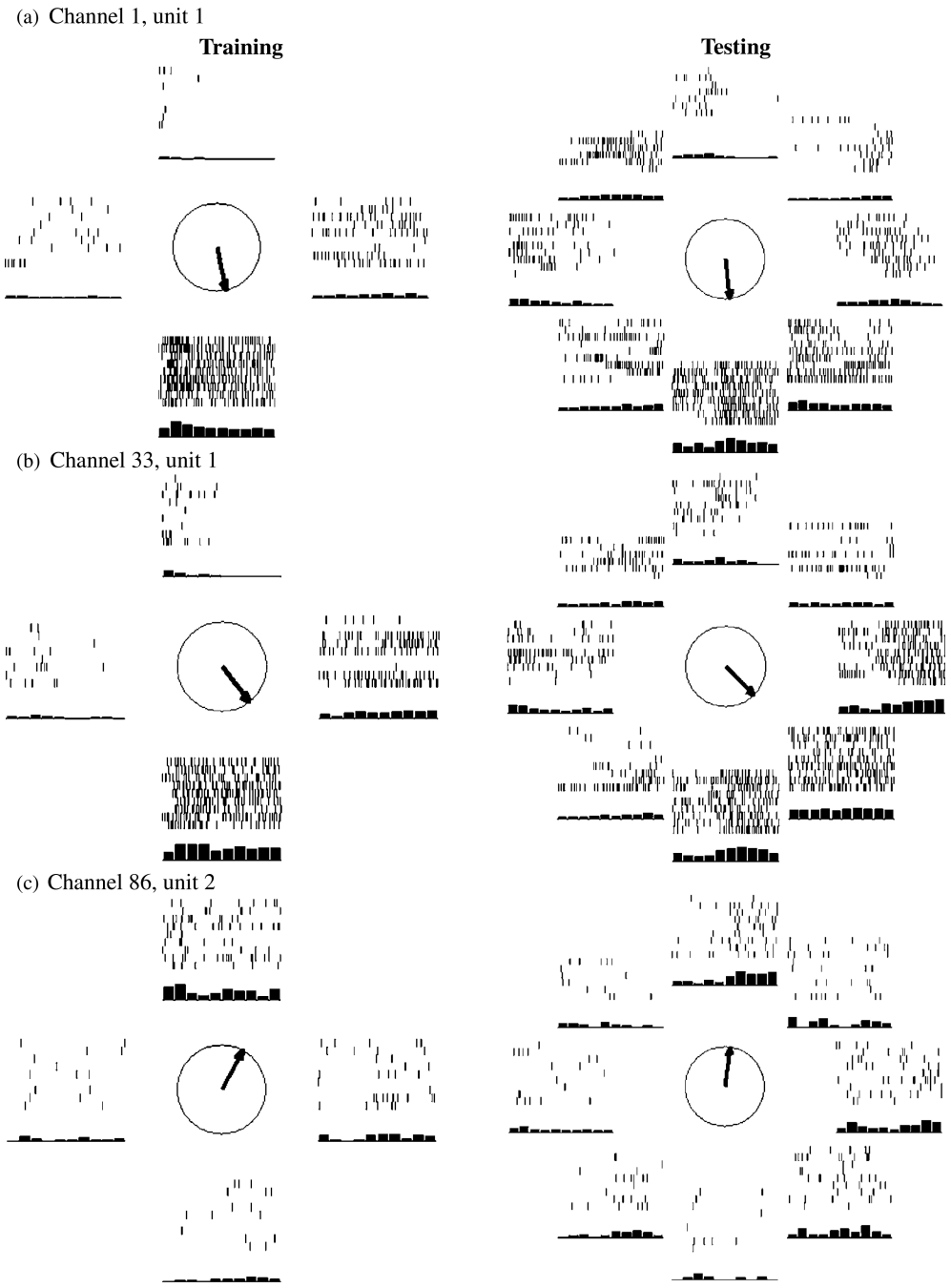

Figure 5.

Consistent directional tuning between training and testing. Sample spike trains are shown for three example units in one session (S3-261), in which S3 performed the center-out-back task during training (left) and testing (right). Black tics in the raster plots show the times when neuronal spikes occurred after target onset. Spike raster plots are arrayed according to the direction of movement (four directions during training and eight during testing). The arrows indicate the preferred directions estimated for each unit. The bars below the spike raster plots show the histogram of spikes for 1 s in 100 ms non-overlapping windows.

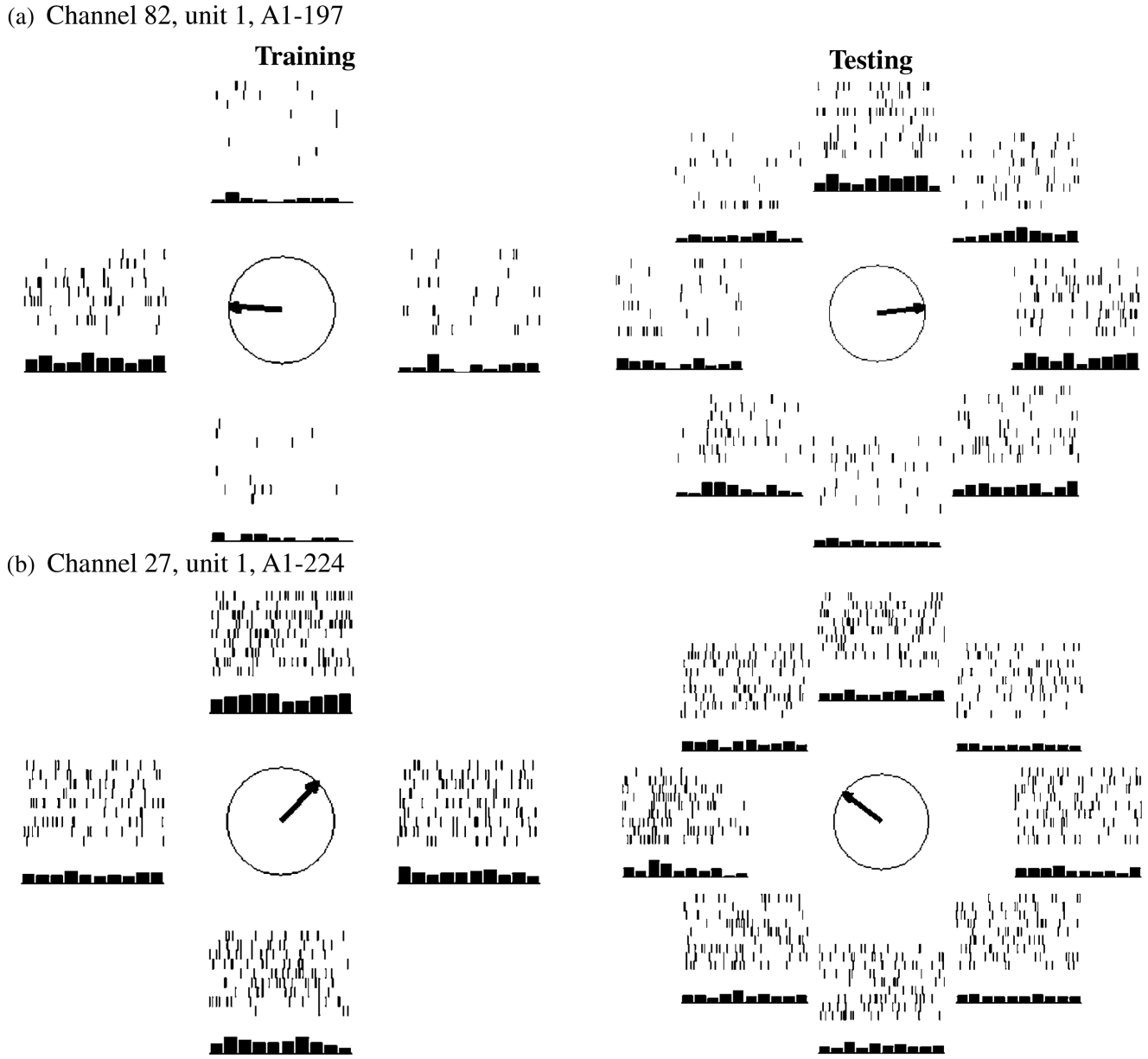

Figure 6.
Changes in directional tuning between training and testing. The spike trains of two sample units from A1-197 (*a*) and A1-224 (*b*) are shown. The left and right columns illustrate tuning during training and testing, respectively. The arrows indicate the estimated preferred direction (PD).

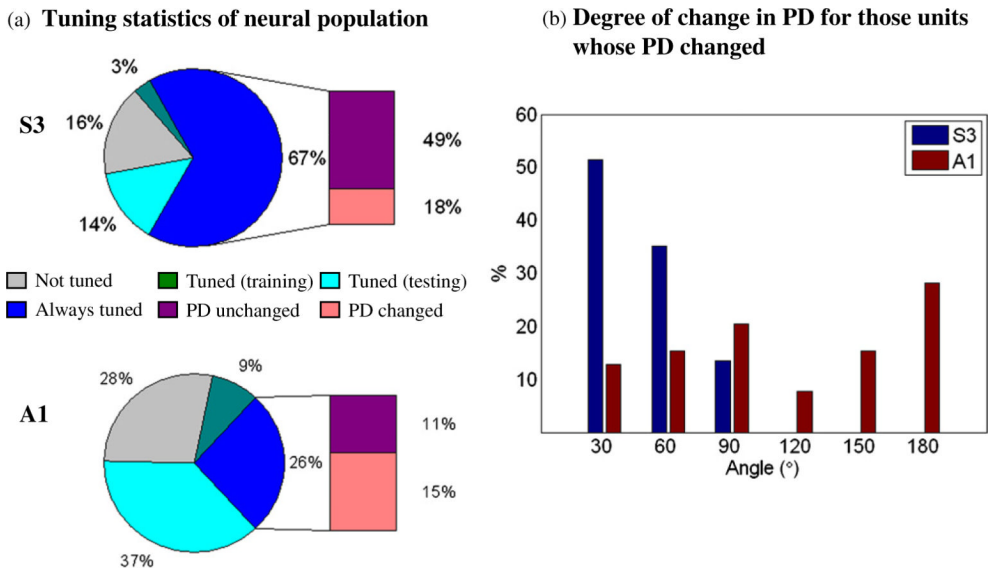

Figure 7.

Statistics of directional tuning change. (*a*) The statistics of tuning across training and testing is illustrated in terms of the percentage of subsets of units with particular properties in S3 (over six sessions) and in A1 (over three sessions). The neural population was divided into four subsets including: units showing no directional tuning across the session, units directionally tuned only during training, units tuned only during testing, and units tuned during both training and testing. The last subset is further divided into smaller groups including: one showing significant PD changes and the other showing no PD change. (*b*) For units that changed PD, the bars indicate the percentage of units that changed by the indicated number of degrees (0–30, 30–60, etc) for S3 (blue) and A1 (red).

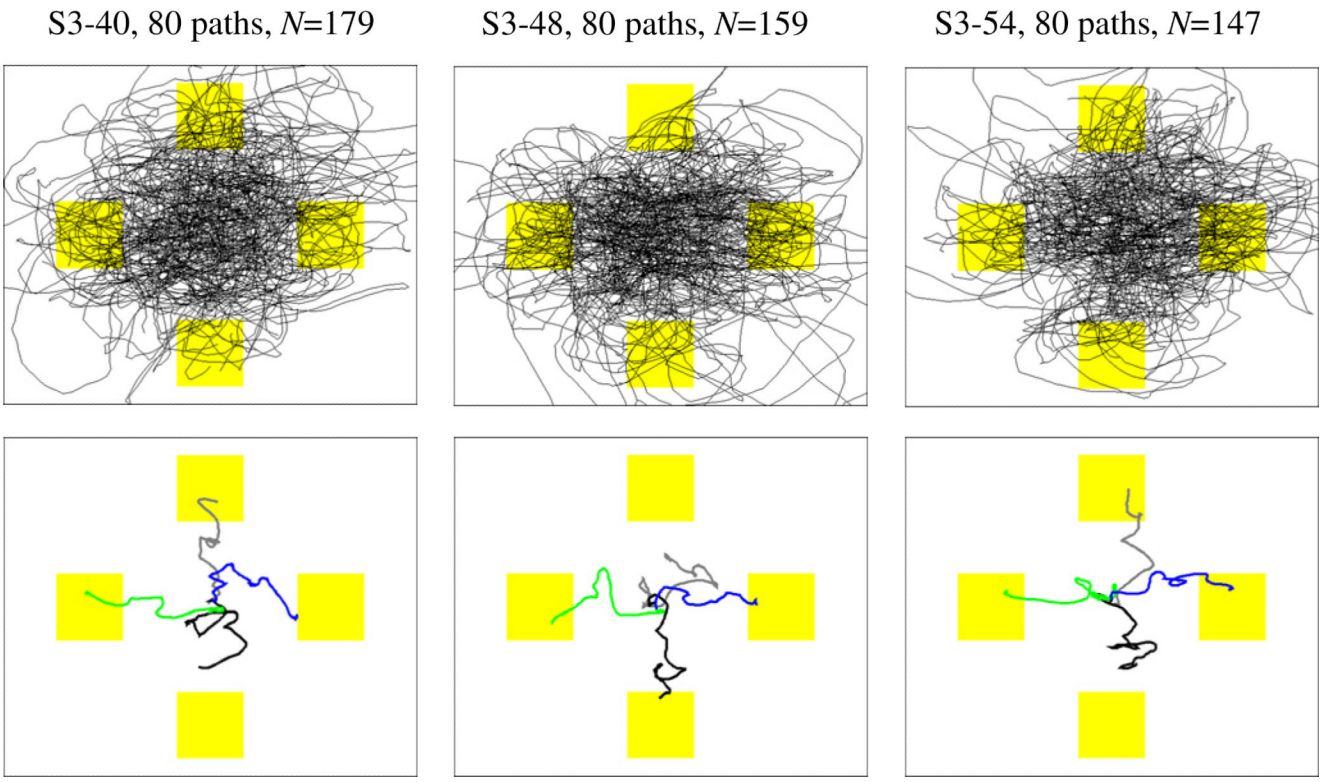

Figure 8.

The NC movement paths decoded by the *position-based linear filter* in S3. The NC movement paths from the center to four radial targets performed by S3 using the position-based linear filter in three recording sessions (S3-40, S3-48 and S3-54). (Top) 80 NC paths (20 per target) denoted by black lines are shown per session. The yellow squares approximately represent target positions and sizes. *N* denotes the number of recorded units. (Bottom) the mean NC path toward each target is shown for each session. Different line colors denote the mean path for different targets: blue: 0°; gray: 90°; green: 180°; black: 270°. To obtain the mean path, all the NC paths per target were linearly interpolated to make each path (with different duration) have the same number of sample points. The mean path per target was then computed over these interpolated paths. Note that some mean paths did not reach the target. This illustrates that for some trials, the NC failed to reach the target before timeout expired.

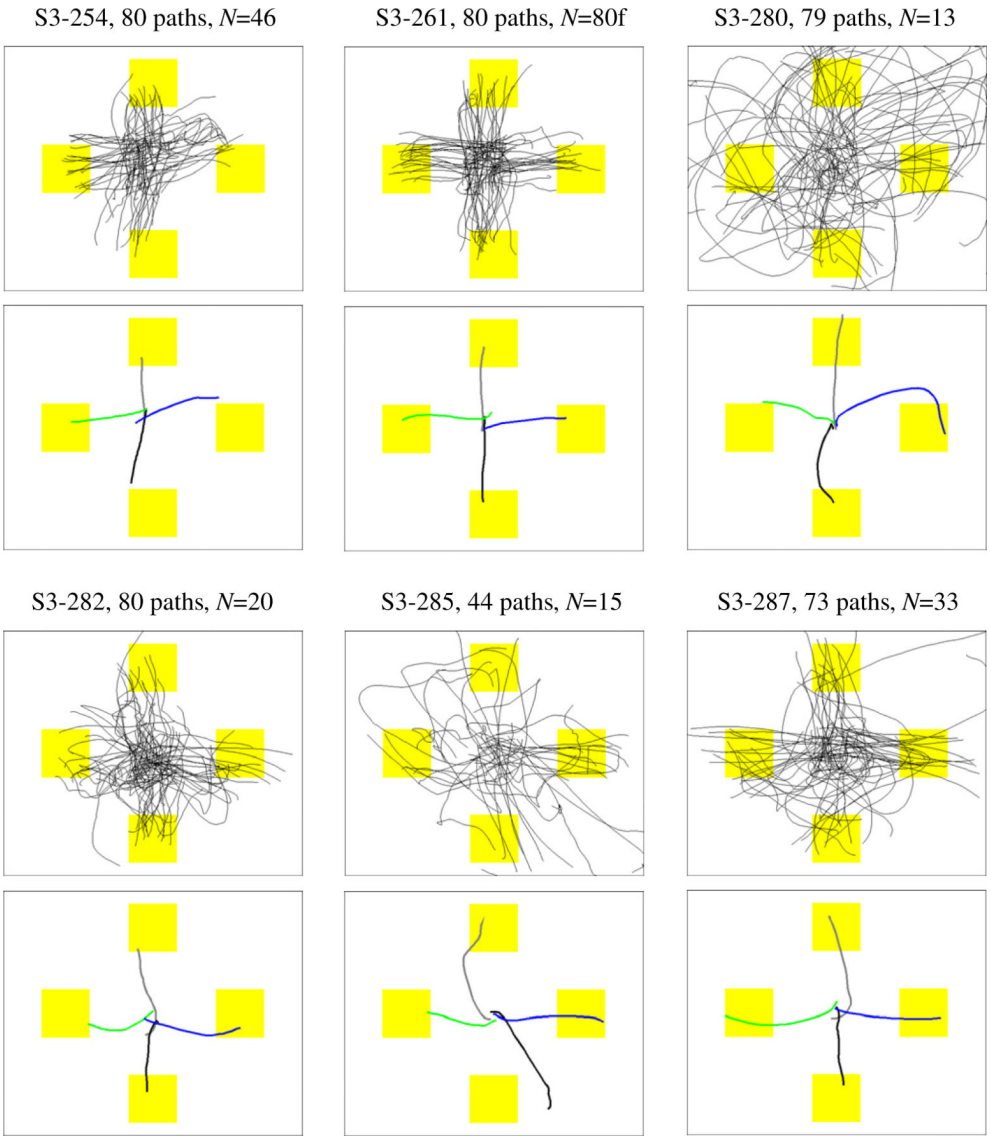

Figure 9.

The NC movement paths decoded by the *velocity-based Kalman filter* in S3. The NC movement paths from the center to four radial targets performed by S3 using the velocity-based Kalman filter in six recording sessions (S3-254, S3-261, S3-280, S3-282, S3-285 and S3-287). 44–80 NC paths (10–20 per target) are shown together with the mean NC paths. *N* denotes the number of recorded units.

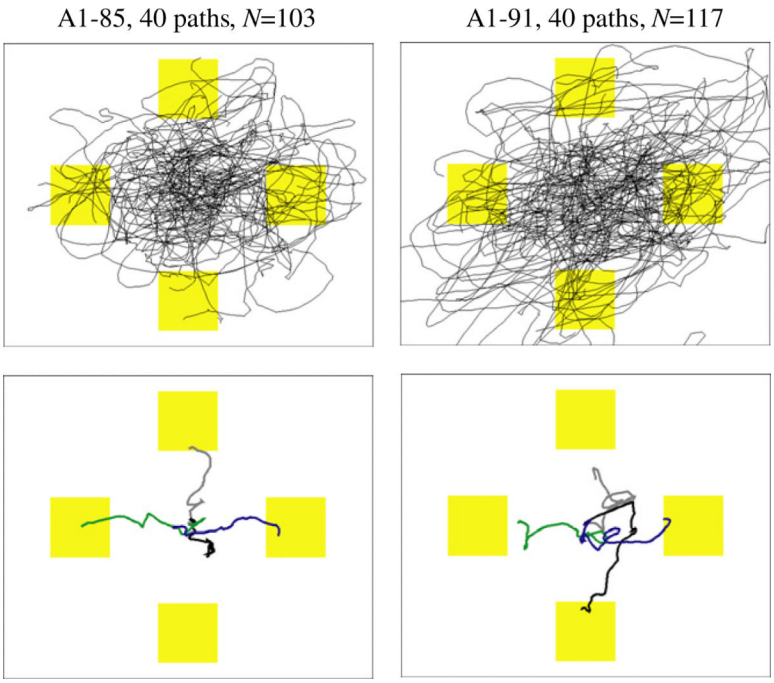

Figure 10.
The NC movement paths decoded by the *position-based linear filter* for A1. The NC movement paths performed by A1 using the position-based linear filter in two recording sessions (A1-85 and A1-91). (Top) 40 NC paths (10 per target) are shown. (Bottom) The mean NC path toward each target is shown for each session.

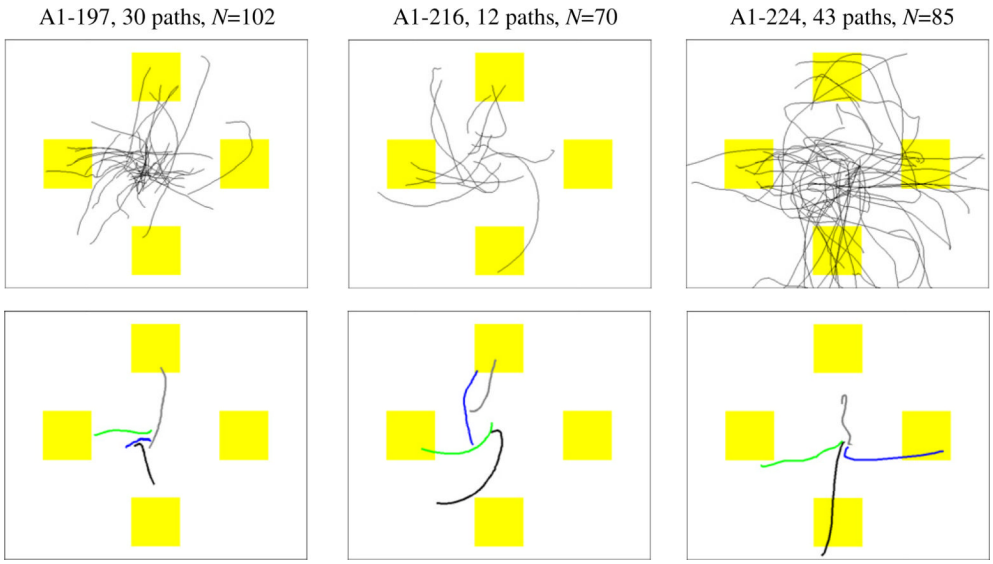

Figure 11.

The NC movement paths decoded by the *velocity-based Kalman filter* for A1. The NC movement paths performed by A1 using the velocity-based Kalman filter in three recording sessions (A1-197, A1-216 and A1-224). (Top) 12–43 NC paths (2–11 per target) are shown. (Bottom) The mean NC path toward each target is shown for each session.

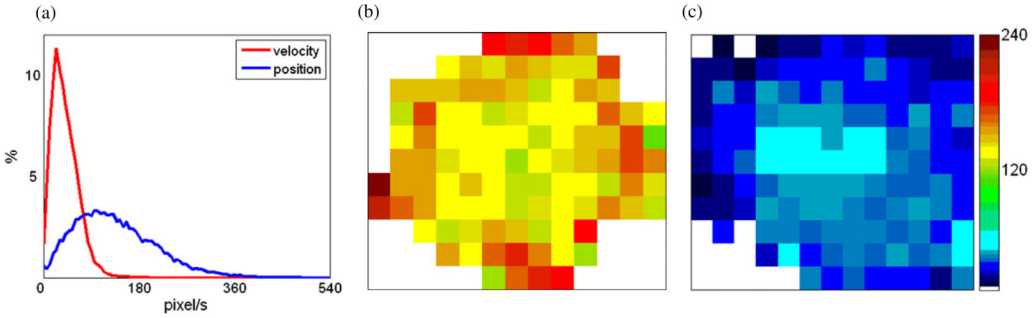

Figure 12.

The distribution of neural cursor speed. (*a*) The NC speed distributions with position-based linear filter decoding (blue) or velocity-based Kalman filter decoding (red) are presented. The NC speed data were collected from nine sessions of S3 with approximately 40 000 time samples for each of position and velocity decoding. (*b*) Mean cursor speed is shown as a function of 2D cursor location for position-based decoding. Each pixel depicts the mean speed of the NC in that region of the screen. The highest cursor speeds are found at the periphery of the movement space near the targets. (*c*) Mean cursor speed for velocity-based decoding is shown; the highest speeds are found in the center of the screen with lower speeds near the targets. For both (*b*)and(*c*), the dotted squares represent the target size and location.

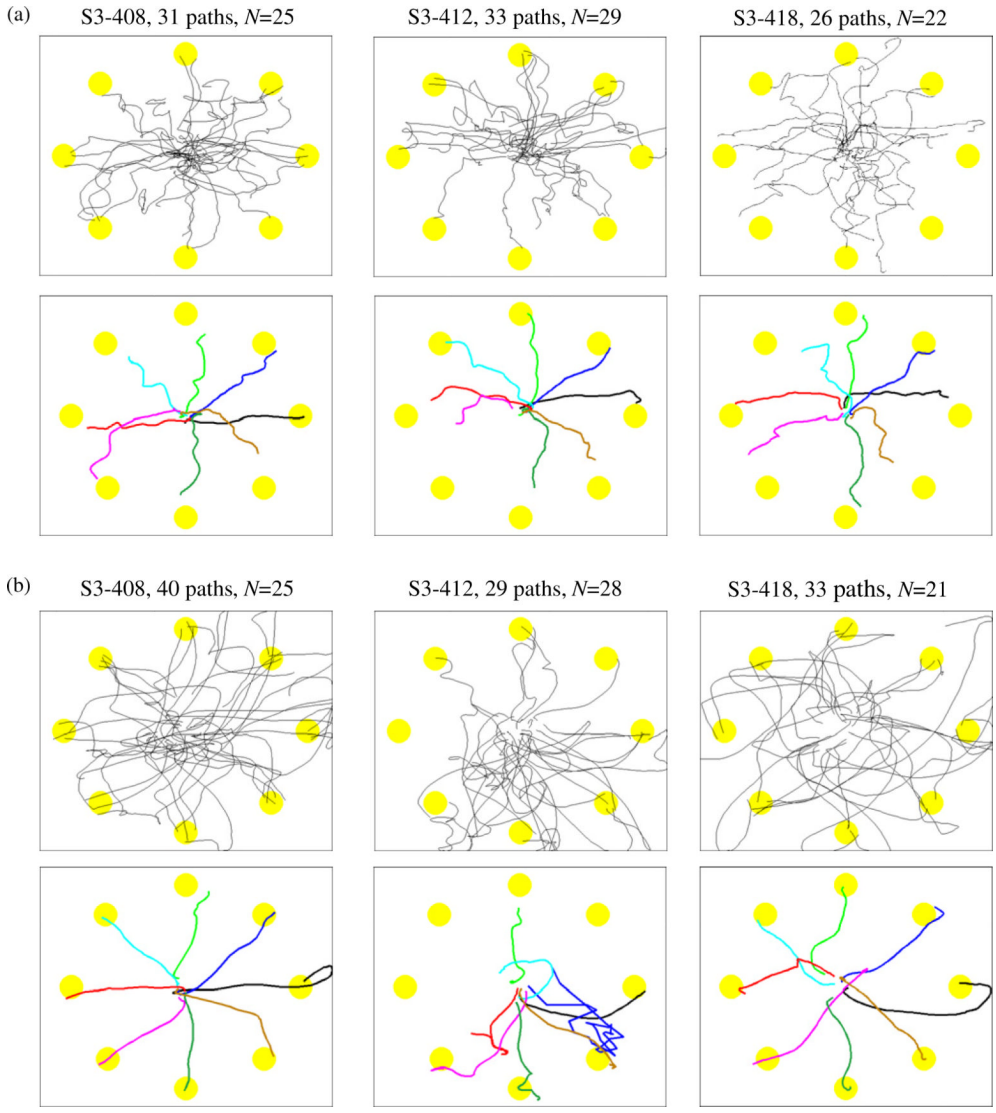

Figure 13.

The NC paths decoded by the velocity-based Kalman filter and linear filter. The NC paths performed by S3 using the velocity-based decoders during the eight-target center-out task are shown. Performance results from three recording sessions (S3-408, S3-412 and S3-418) are shown where both the linear filter and the Kalman filter were sequentially tested. The individual NC paths and the mean path to each target made using either (*a*) the linear filter or (*b*) the Kalman filter are shown. Eight targets were located at (0°,45°,90°, 135°, 180°, 225°, 270° and 315°). *N* denotes the number of units used to build the decoding filter.

Table 1

Categorization of the recording sessions based on decoding algorithms and training task.

Session group label: kinematic model, decoding method, task	Session index (unit count)	Decoding algorithm	Training task
Position, linear filter, pursuit tracking (PLP)	S3-40 (179), S3-48 (159), S3-54 (147); A1-85 (103), A1-91 (117)	Position-based linear filter	Random pursuit-tracking task
Velocity, Kalman filter, center out (VKC)	S3-254 (46), S3-261 (80), S3-280 (13), S3-282 (20), S3-285 (15), S3-287 (33), A1-197 (102), A1-216 (70), A1-224 (85)	Velocity-based Kalman filter	Center-out-back task
Velocity, linear/Kalman, center out (VLKC)	S3-408 (25), S3-412 (28), S3-418 (21)	Velocity-based linear and Kalman filters	Center-out-back task

Table 2

Neural cursor control evaluation in two participants (S3 and A1) in performance of the four-target center-out task (see section 2.4 for the description of the testing task). *N* denotes the number of recorded neuronal units and N_c denotes the number of units that were significantly correlated with a given decoding kinematic variable (position or velocity; see figure 3). The significant target acquisition rate (STAR: target acquisition rate – control rate) is also reported. The control rate was 9.8% for position decoding and 3.9% for velocity decoding.

						ODC		MDC		ME (mm)		MV (mm)	
Trial day	*N*	N_c	MT (s)	STAR (%)	ER (%)	Mean	SD	Mean	SD	Mean	SD	Mean	SD
S3—position-based linear filter decoding													
40	179	126	2.87	67.7	22.5	10.1	6.3	9.6	5.5	33.7	14.5	32.7	16.7
48	159	120	2.98	62.7	27.5	10.3	7.2	10.2	4.9	32.6	18.0	33.3	16.5
54	147	110	3.06	75.2	15.0	9.3	6.2	10.0	6.2	32.8	13.0	31.8	14.0
Average	162	118	2.97	68.5	21.7	9.9	6.5	9.9	6.2	33.0	15.1	32.4	15.6
S3—velocity-based Kalman filter decoding													
254	46	40	3.71	74.8	21.3	0.6	0.7	1.6	1.8	20.0	11.5	8.8	6.0
261	80	65	3.29	92.3	3.8	0.5	0.6	1.9	1.6	17.1	9.8	7.9	4.3
280	13	8	3.47	72.0	24.1	0.8	1.0	1.4	1.2	34.3	24.2	19.7	11.7
282	20	13	3.71	82.3	13.8	0.8	0.9	1.9	1.5	21.2	11.3	12.6	7.1
285	15	13	3.02	81.1	15.9	1.2	1.8	1.8	1.6	25.8	19.6	14.8	98
287	33	27	3.40	92.0	4.1	0.7	0.9	1.6	1.4	23.0	15.3	14.3	8.9
Average	35	28	3.46	82.3	13.8	0.7	1.0	1.7	1.5	23.0	16.3	12.7	8.9
A1—position-based linear filter decoding													
85	99	60	2.88	71.4	18.8	9.4	5.3	9.7	4.9	27.4	9.4	26.8	10.1
91	112	72	3.55	55.2	35.0	10.2	6.7	11.3	6.0	42.5	23.0	41.1	22.9
Average	106	66	3.18	63.9	26.3	9.7	5.9	10.4	5.5	34.1	18.4	33.1	18.3
A1—velocity-based Kalman filter decoding													
197	102	68	3.89	39.4	56.7	0.9	1.6	1.6	1.9	16.9	8.3	13.2	11.4
216	70	43	3.95	62.8	33.3	0.8	1.0	2.0	1.8	21.0	8.7	12.6	7.7
224	85	52	2.71	82.2	13.9	0.9	1.1	2.0	1.5	29.1	20.1	13.7	8.5
Average	94	60	3.15	64.3	31.8	0.9	1.2	1.9	1.6	25.3	17.5	12.8	7.6

Table 3

Neural cursor control performance evaluation in S3 for the eight-target center-out task using velocity-based decoders. In each of three recording sessions, two velocity-based decoders were compared: the linear filter (LF) and the Kalman filter (KF). *N* is the number of neuronal units used to build the decoders.

Trial day	*N*		MT (s)		ER (%)		ODC		MDC		ME (mm)		MV (mm)	
	LF	KF	LF	KF	LF	KF	LF	KF	LF	KF	LF	KF	LF	KF
408	25	25	7.80	5.79	35.5	10.0	2.3(1.8)	1.3(1.4)	5.4(1.6)	2.8(1.7)	11.0(7.6)	14.9(8.6)	8.4(3.7)	11.4(5.8)
412	29	28	7.51	5.88	39.4	44.8	1.3(1.3)	1.1(1.4)	4.3(2.1)	2.9(1.7)	13.3(6.3)	14.5(7.4)	8.0(2.7)	11.0(4.3)
418	22	21	8.04	6.42	76.9	12.1	2.8(1.7)	1.3(1.2)	5.8(2.6)	2.9(1.6)	11.4(5.5)	18.8(9.2)	8.6(2.7)	14.4(6.9)
Avg	25	25	7.71	6.01	48.9	26.0	1.9(1.7)	1.3(1.3)	4.9(2.0)	2.9(1.7)	12.1(6.7)	16.0(8.5)	8.2(3.1)	12.1(5.8)