nature machine intelligence

Supplementary information

https://doi.org/10.1038/s42256-025-01090-y

Brain–computer interface control with artificial intelligence copilots

In the format provided by the authors and unedited

Contents

1 Additional Tasks Details

Open Loop Text (referred to as the "open loop" task). We refer to the presentation of one action as a "trial." Each trial was 20 seconds long. The participant was instructed to repetitively perform an action corresponding to the presented text. There was a 2 second inter-trial interval. During the inter-trial interval, a red box was displayed and the participant was asked to relax and not to perform any action.

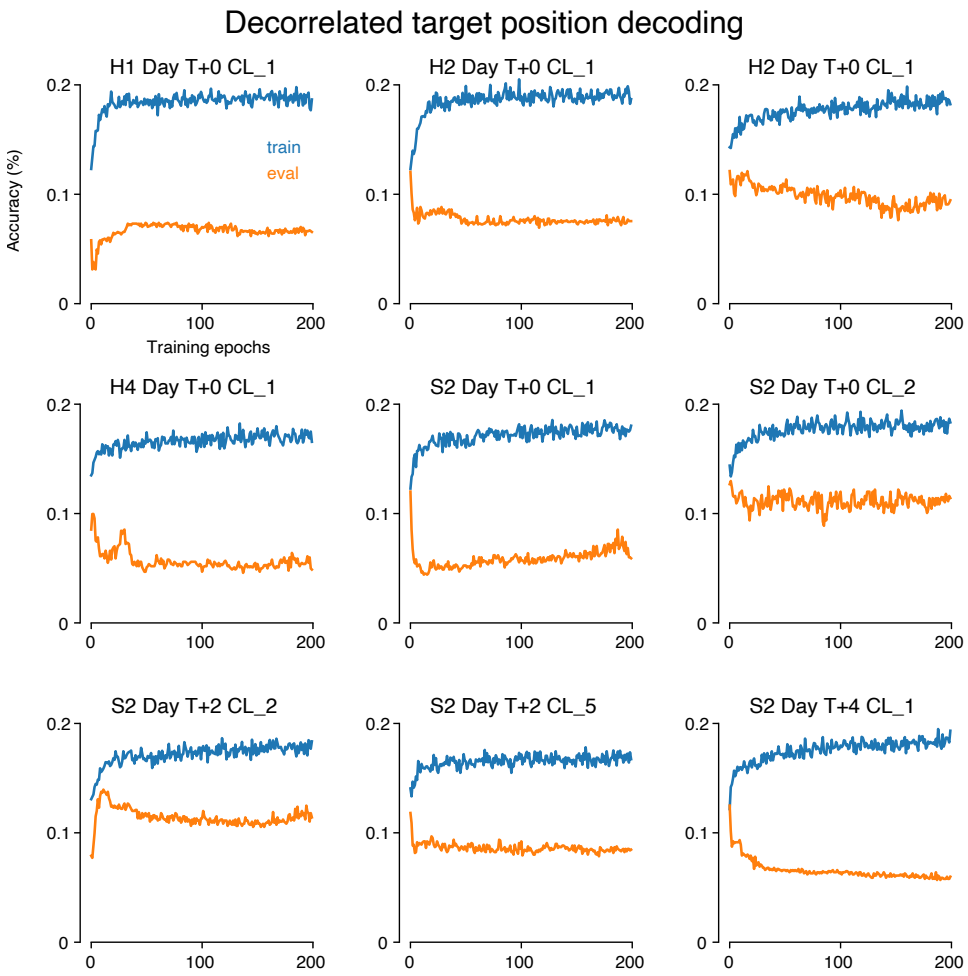
Decorrelated Closed Loop (referred to as the "decorrelated" task). This task utilized a decoder trained from the open loop task that controlled the position of a cursor. We provided the user with real-time feedback of the cursor position.

At the start of a decorrelated trial, a cursor was displayed at the center of the screen. A text prompt (from the same set used in the open loop task) appeared inside a rectangular box at one of eight target locations (from 0° to 315° in 45° increments, centered at a radius of 9.34 cm and width of 4.67 cm). The text prompt and its target location were uncorrelated. For example, 'Left Leg' could appear at any of the eight locations for that trial. The output from the decoder trained on the open loop data was used for the cursor's movement in the decorrelated session. The decoder was continuously adapted (see Methods 1.7 for more details). The cursor's movement was restricted to move along the straight line segment connecting the center of the screen to the outer edge of the target. To do this, we decoded 2D velocity and rotated the velocities to be congruent so that the correct action moved the cursor towards the correct target. For example, if 'Left Leg' was at the up right target (45°) then any left cursor movements moved the cursor up right. The 2D velocities were subsequently projected onto the straight line segment connecting the center of the screen to the outer edge of the target. Intuitively, the cursor moved toward the target if the decoded action was the same as the prompted action. The cursor moved away from the target if the decoded action was opposite.

Although a target was presented in the decorrelated closed loop task, the target was not selectable, and remained onscreen for 7 seconds, with cursor movement occurring only in the last 5 seconds. This allowed for an even distribution of target positions. We chose to not make targets selectable to acquire an equal number of samples per class in the training data. If targets were selectable, then actions that were more accurately decoded would be underrepresented in the decorrelated task data. If the decoder continued to decode the correct action, the cursor would hover within the target acceptance window for the duration of the trial.

This task decorrelates the target's physical location (and therefore eye movements) from motor intent. A performant decoder must *ignore* eye artifact, since eye movement signals more frequently correspond to classes other than the correct class, and would be detrimental to the decoder. Performant decoders trained using data from the decorrelated task actively learn to minimize the influence of eye position (Supplementary Figure 1).

Center-out 8 Target Selection (referred to as the "center-out 8" task). During the center-out 8 task, the participant controls a cursor to acquire 8 targets equally spaced on the circumference of a circle. The cursor was displayed at the center of the screen in grey color. The position of the cursor was bounded to a square with x-position between $[-1, 1]$ units and y-position between $[-1, 1]$ units (total side length: 2 units). 1 unit corresponded to 11.675 cm. The target appeared in green color at one of the 8 locations spaced 0.7 units from the center at 45° intervals. The target diameter varied from 0.6 units (7 cm) to 0.2 units (2.3 cm). To acquire a target successfully, the decoded cursor had to be held contiguously over the target for 500 ms. The trial timed out if the participant was not able to acquire the target within 24 seconds. After each center-out target, the next prompted target was at the center of the screen. If the participant wasn't able to acquire the center target, the cursor was reset to the center and the target was placed at one



Supplementary Figure 1. Target position was not reliably decoded from EEG activity during the 1D decorrelated closed-loop task. Training accuracy (blue) and validation accuracy (orange) for decoding *target position* from the CNN output hidden state in all decorrelated training sets. Note that we recalibrated S2's CNN on two separate days (labeled T+2, T+4), which is why S2 has multiple decorrelated sessions. Eye movements and motor actions are decoupled in the decorrelated task, meaning that CNN features that decode motor intent should not decode target position (e.g., what target the user is either looking at, or following the cursor to) above chance. This reflects that eye movement artifacts are not represented in the CNN hidden state. Consistent with this, we did not observe consistent above chance (12.5%) decoding target location from the CNN hidden state. After training, we often achieved worse than chance, indicating overfitting. We therefore show the validation accuracies across all epochs of training to demonstrate that validation accuracy does not achieve above chance accuracies throughout training.

of the center-out locations. If a trial ended with target acquisition, it was counted as a success. If the trial ended after timeout, it was counted as a failure. Statistics for the center-out 8 task are only computed on the 8 center-out trials to be agnostic to target location and are averaged over each set of 8 trials.

2 Additional experimental protocol details

2.1 Participant selection

Two additional SCI paraplegic participants were evaluated for a single day, but did not achieve high decoding performance for attempted leg movements. This does not mean these participants could not

control a BCI, but only that they did not exhibit substantial EEG modulation for the prompted behaviors. To be clear, we did not perform further experiments to identify a behavior set that could produce 4 decodable classes, since determining these behaviors requires an open-ended search over potentially many experimental sessions, a tangential question representing future work to make EEG BCIs more robust (see Discussion). Two additional participants (one SCI, the other healthy) conducted pilot experiments, achieving 2D cursor control, but did not complete experiments for personal reasons unrelated to the study. Dates of the 5 experimental sessions (Figure 3) are shown in Supplementary Table 1.

	1	2	3	4	5
S2	2024-02-12	2024-02-13	2024-02-21	2024-02-27	2024-03-18
H1	2024-02-13	2024-02-14	2024-02-15	2024-02-20	2024-02-21
H2	2024-02-02	2024-02-05	2024-02-06	2024-02-07	2024-02-09
H4	2024-03-01	2024-03-04	2024-03-06	2024-03-11	2024-03-13

Supplementary Table 1. Dates of experimental sessions 1-5 to collect data from the participants.

2.2 Exploring Alternative EEG Modulation Strategies

To maximize EEG decoding performance, one could search for sets of motor or cognitive actions that best modulate EEG for each participant. Because we were focused on shared autonomy, in our experiment, we restricted ourselves to participants who had modulation for our selected actions. In paralyzed participants, we only tested the left leg, right leg, and both leg movements for EEG modulation. If a participant did not demonstrate strong modulation for these actions, we did not perform any further experiments. But we note that this does not mean these participants would be unable to control a BCI. In particular, they may have modulated EEG activity for other actions. Several studies have explored diverse action sets for EEG including seating, standing[1], individual finger flexion[2], and cognitive tasks such as word generation and cube rotation[3]. For participants whose EEG do not modulate well for leg actions, future work may identify alternative actions with more salient features to reach baseline decoding performance.

3 Additional EEGNet (CNN) training details

Custom Python code was used to implement and train the EEGNet. We split the data into 5 folds at the trial level, and performed 5-fold cross-validation over these folds. The learning rate was set to 10^{-3} at the start, and reduced the learning rate by a factor of 0.1 if the accuracy on the held-out validation fold did not improve for 10 consecutive epochs. We also halted training if the minimum validation loss did not improve for 10 consecutive epochs. From the 5-fold cross-validation, we chose the model for online control based on the validation confusion matrix and overall accuracy.

4 Additional cursor copilot details

4.1 Network architecture and training hyperparameters

The value, $V(\mathbf{s})$, and policy, $\pi(\mathbf{s})$, networks were composed of a two-layer multi-layer perceptron (MLP) with 64 hidden units, followed by an LSTM with a 256-dimensional hidden state. The copilot was trained using PPO with a fixed learning rate of 3×10^{-4} with a rollout buffer size of 2048 and a batch size of 64. Copilots were trained for a total of $1,200,000$ steps.

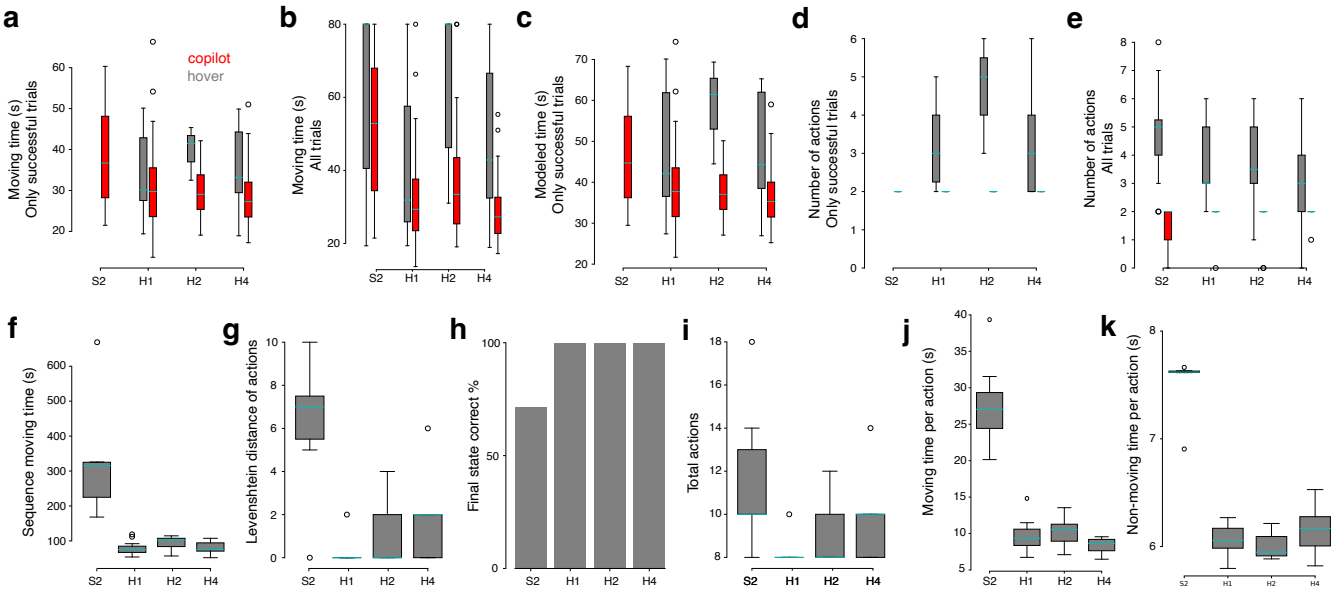
5 Additional robotic arm and computer vision copilot details

We interfaced with the Franka Emika Panda 3 robotic arm using the frankalib and liborl C++ libraries with custom C++ and Python code, mounted above a table. A camera (Intel RealSense D455, 1280x800px, 30Hz, RGB) was positioned with a tripod such that it viewed the working space of the robotic arm. We calibrated the computer vision system using 4 marked locations on the table, and masked the image corresponding to the working space. A separate process used GroundingDINO[4] to find the pixel locations corresponding to the "small cube" and "cross" text prompts. This typically took 350-400ms on an RTX A4000 and can be extended to arbitrary objects. For safety, the robot's (x,y) position was limited to be between the polygon defined by vertices $(0.3,-0.31)$m, $(0.6,-0.31)$m, $(0.75,-0.15)$m, $(0.75,0.15)$m, $(0.6,0.31)$m, and $(0.3,0.31)$m.

A 5-second cooldown for initiating a new pick/place action was enforced after the end of each pick/place action. The 4 blocks and 4 target crosses were placed at $[(0.675,-0.179)$, $(0.674,-0.058)$, $(0.672,0.075)$, $(0.674,0.207)]$m and $[(0.387,-0.174)$, $(0.388,-0.052)$, $(0.392,0.087)$, $(0.395,0.210)]$m and were reset in between trials (12-14cm between adjacent blocks/targets, and 28cm between each block and the closest target). The robotic arm's end effector was reset to $(0.3,-0.021,0.081)$m before each trial. The z-position of the robotic arm was approximately $(-0.03+0.11\cdot\min(5\cdot(d-0.0254),1))$m, with the distance to closest eligible block/target d in meters. Each trial began when a button to allow movement was pressed and ended when the user placed a block at any location or when 40 seconds of moving time (time not dedicated to a pick/place action) accumulated.

As with the paired pick-and-place task, during the sequence pick-and-place task, the robotic arm automatically picked up or placed blocks whenever its (x,y) position was within 2.54 cm of the detected center of an available goal. On earlier trials for S2 sequence task (4/7 trials), the maximum speed was only 4 cm/s, and the robotic arm only picked/placed the block when the end effector was within 2.54 cm of the block/target and the block/target had the highest score $=(r_{target}-r_{robot})\cdot\angle(v,r_{target}-r_{robot})$, where $\angle(v,r_{target}-r_{robot})$ is the angle between the 2D KF velocity v and 2D displacement $r_{target}-r_{robot}$. This resulted in 2 instances on 2 trials where the robotic arm failed to place a block on a target which it entered the radius of (time difference 18.4 s, 10.7 s) and 1 instance on the first of the aforementioned trials where the robotic arm failed to pick up the correct next block in the sequence and instead picked up an incorrect block (time difference 5.3 s). Since these instances decreased performance, we did not modify statistics for these trials.

We show additional metrics for robotic arm tasks in Supplementary Figure 2 and Supplementary Table 2.



Supplementary Figure 2. Moving time and other metrics favor the computer vision copilot during robotic arm tasks. a, Moving time (excluding time required for raising/lowering arm and closing/opening the gripper) for the paired pick-and-place task for only successful trials. This moving time is a conservative estimate of the copilot improvement, since the copilot helps to successfully grasp and place blocks. In this example, any time spent trying to pick and place blocks without the copilot is excluded, since this only includes time spent moving towards pick and place locations. Note, S2 was unable to perform any trials successfully using only hover, which is why there is no data for the hover condition for S2. Box plots show Q1, median, and Q3. Whiskers show the minimum and maximum of data points that are within [Q1-1.5IQR, Q3+1.5IQR], and outliers are shown outside of that range. Values of *n* (number of trials) are listed in Supplementary Table 2. **b,** Moving time for all trials (paired task). The moving times for trials without a successful pick or without a successful place were set to 80s and 40s, respectively. **c,** This estimates the total trial time in the Paired Pick-And-Place task if every action took 4 seconds to attempt or complete. This estimate, in general, underestimates the actual trial time for hover trials. Nevertheless, the copilot still improves overall time spent to perform the task. **d,** Number of pick and place attempts on eventually successful trials. For the copilot, all successful trials had one successful pick and one successful place. For hover trials, the median number of actions in successful trials was generally higher due to failed pick and place attempts. **e,** Same as **c**, but including failed trials. Note that the number of actions can be less than 2 if zero or one attempted action was made throughout the trial. **f,** Distribution of moving time for the sequence pick-and-place task. **g,** Levenshtein edit distance between prompted action sequences and actual action sequences. The Levenshtein edit distance measures the total number of insertions, deletions, and substitutions required to change one sequence to another. In this instance, it is the minimum number of actions in the user's performed action sequence that must be inserted, deleted, or substituted to produce the prompted action sequence. **h,** Percentage of sequences in which users placed all blocks on the correct target at the end of the trial. Even though there were errors throughout each trial (e.g., placing a block on the wrong cross), these errors could be corrected (e.g., picking up the block from the wrong cross and placing it on the correct cross). **i,** Total number of actions (adjusted to include missing actions for incomplete trials) for all sequence task trials. The minimum number of actions to successfully pick and place 4 blocks at the correct location is 8. **j,** Moving time per executed action (sequence task). **k,** Non-movement time per executed action (sequence task). This characterizes the amount of time between pick/place initiation and successful execution.

Panel	Description	n=[hover, copilot] trials			
		S2	H1	H2	H4
a	Moving time (successful only)	[0, 10]	[10, 25]	[3, 8]	[12, 30]
b	Moving time (all trials)	[32, 32]	[32, 32]	[16, 16]	[32, 32]
c	Modeled time (successful only)	[0, 10]	[10, 25]	[3, 8]	[12, 30]
d	Number of actions (successful only)	[0, 10]	[10, 25]	[3, 8]	[12, 30]
e	Number of actions (all trials)	[32, 32]	[32, 32]	[16, 16]	[32, 32]
f	Sequence moving time	7	15	10	10
g	Levenshtein distance of actions	7	15	10	10
h	Final state correct	average	average	average	average
i	Total actions	7	15	10	10
j	Moving time per action	7	15	10	10
k	Non-moving time per action	7	15	10	10

Supplementary Table 2. Statistics for Supplementary Figure 2. For the paired pick-and-place task, one trial consists of picking up a single block and placing it at a target. For the sequence pick-and-place task, one trial consists of picking up 4 blocks and placing them at 4 targets in a prompted sequence.

125

References

1. Chaisaen, R. *et al.* Decoding EEG Rhythms During Action Observation, Motor Imagery, and Execution for Standing and Sitting. *IEEE Sensors Journal* (2020).

2. Sun, Q., Calvo Merino, E., Yang, L. & Van Hulle, M. M. Unraveling EEG correlates of unimanual finger movements: insights from non-repetitive flexion and extension tasks. *Journal of NeuroEngineering and Rehabilitation* **21**, 228 (2024).

3. Rosanne, O., de Oliveira, A. A. & Falk, T. H. EEG Amplitude Modulation Analysis across Mental Tasks: Towards Improved Active BCIs. *Sensors* **23**, 9352 (2023).

4. Liu, S. *et al.* Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection. *arXiv preprint arXiv:2303.05499* (2023).