2023

Brain Computer Interface

PHD PRELIMINARY EXAM

MEHDI KHANTAN

TEMPLE UNIVERSITY

List of Figures

Figure 1 - Three types of prominent BCI sensors 3

Figure 2 - Sensors of three different BCI technique 4

Figure 3 - The Black Rock MEA 8

Figure 4 - The blocks of two types of data gathering sessions. 9

Figure 5 - The schematic of the system and process of supervised single-characters detection. .. 10

Figure 6 - The screen of supervised sentences detection 11

Figure 7- The block diagram of the system 12

Figure 8 - TWPCA process for one character with 27 repetitions 13

Figure 9 - The HMM Viterbi process for time bins of neural data threshold crossing. 15

Figure 10 - The block diagram of the Real Time Decoder 17

Figure 11 - Architecture of GRU. 18

Figure 12 - The RNN architecture. 19

Figure 13 - The output of the GRU-RNN 21

Figure 14 - The character error rate for the GRU-RNN 23

Figure 15 - Character error rate. 25

Figure 16 - Intended pen trajectory 25

Figure 17 - Error rate comparison between using a shared and day specific input layer. 26

Figure 18 - A block diagram of the system and its various sections. 29

Figure 19 - Overhead view of ARAT (A) and Object Transform (B) tests 30

Figure 20 - Torque-to-force conversion example and electrode characteristics. 33

Figure 21 - Experimental timeline 36

Figure 22 - Effect of ICMS feedback on individual ARAT task item completion speeds. The * are showing the average of all the trials. 37
Figure 23 - The Results of ARAT test. 39
Figure 24 - A Schematic diagram of the experiment. 41
Figure 25 - Real-time control of 1D hand movements. 48
Figure 26 - Real-time control of 3D hand movements. 50
Figure 27 - Neuro Dropping Analysis. 52

List of Tables

Table 1 - Transition probabilities for the HMM Model 14
Table 2 - Starting point of characters divided to four groups. 16
Table 3 - Error rates of the HMM and the GRU-RNN in this study 22
Table 4 - Summary of the performance. 27
Table 5 - The results of ARAT test 38

Table of Content

List of Figures I

List of Tables II

1 Introduction 1

1.1 Building Blocks of Brain Computer Interface 2

1.2 Types of Brain-Computer Interfaces 3

2 High-performance brain-to-text communication via handwriting 8

2.1 Introduction 8

2.2 Data Collection Method 9

2.3 Signal Processing 11

2.4 Time Warping 13

2.5 Hidden Markov Model 14

2.6 HMM Viterbi 15

2.7 Synthetic Data Generation 16

2.8 Real Time Decoder 17

2.9 RNN Architecture 18

2.10 Comparison of GRU-RNN to an HMM decoder 21

2.11 Decoder retraining analysis 22

2.12 Adding Noise 23

2.13 Language model 24

2.14 Unsupervised Training 25

2.15 Pen trajectory visualization 26

2.16 Performance 27

3 A brain-computer interface that evokes tactile sensations improves robotic arm control 28

3.1 The participant 28

3.2 The Micro Electrode Arrays 28

3.3 Prosthetic Limb 30

3.4 Performance Analysis 30

3.5 Data Gathering 31

3.6 Intracortical microsimulation 31

3.7 Decoder Design 34

3.8 Results 38

3.9 Conclusion 40

4 Real-time prediction of hand trajectory by ensembles of cortical neurons in primates 40

4.1 Implantation Procedure 41

4.2 Tasks 42

4.3 Data Analysis 43

4.4 Real Time Data Analysis 47

4.5 Artificial neural network model 47

4.6 Results 49

5 Conclusion 54

6 References 56

1 Introduction

Neurological disorders have substantial societal and economic impacts [1]. In the United States alone, there are almost 7 million stroke survivors [2], of whom 62% have lost mobility in their upper extremities (UE) [3]. Other disabilities include spinal cord injury (SCI), which has affected 291,000 individuals in the United States [4]. Neurological disorders can result in a persistent inability to conduct Activities of Daily Living (ADLs) independently [4–6]. Stroke [2,7,8] and SCI [4,9,10] usually result in debilitating motor deficits of the UE that persist beyond rehabilitation discharge [4,8]. Individuals with moderate to severe neurological UE impairment, for instance, frequently exhibit limited active movement in their paretic elbow and little to no active movement in their paretic arm, wrists, and fingers [11,12].

Using assistive technologies and equipment, the quality of life for stroke patients can be improved. A plethora of intelligent assistive devices and technologies have been developed for individuals with a wide variety of impairments. Most of these solutions rely on residual motor skills or speech recognition. In contrast, brain computer interface (BCI) systems bypass motor output by measuring and decoding neural activities, such as specific attention, imagined movement, or speaking attempts, thus granting patients a degree of autonomy [13]. BCI has developed as an alternative communication channel between the human brain and output devices by enabling the communication between the brain and external devices, like a computer or prosthetic limb. Typically, this is accomplished by recording the brain's electrical activity and then using machine learning algorithms to decode neural signals and transform them into real time commands for an external device.

BCI applications were initially designed to enable individuals with disabilities in communicating, operating computers, and using assistive devices such as wheelchairs, wearable exoskeletons, and

robotic arms to assist patients with paralysis, amputations, and other causes of loss of central nervous system functions. Historically, BCI research has focused on assisting individuals with open-loop systems designed to decode low-degrees-of-freedom movement via neural activities. Due to recent advances in machine learning and intracortical microstimulation techniques, researchers can develop BCI systems with additional degrees of freedom and tactile feedback. In this document, we have investigated three different papers on BCI, in which the first paper is one of the most recent studies in this field that has been done on a person with an SCI and limited hand movement ability, aiming to decode 31 characters by his imagining in writing the character with a pen on paper; the second study aims to provide a prosthetic limb movement with tactile feedback for an individual with tetraplegia; and the third paper is one of the pioneering studies in this field, attempting to decode the three-dimensional hand and arm movements of monkeys through BCI.

1.1 Building Blocks of Brain Computer Interface

All BCI systems employ a data and signal flow consisting of the following phases.

1- Signal acquisition: Neural activity is captured by means of electrodes placed in, on, or near the brain. The electrode type and placement significantly constrain the types of neural signals that can be captured.

2- Signal processing: The raw signals are processed to retrieve pertinent information. Typically, this involves filtering the signals to eliminate potential noise and employing feature extraction techniques to reduce dimensionality.

3- Machine learning: The extracted features are then classified into different categories or states, such as different types of mental tasks or motor movements, using machine learning or pattern recognition algorithms.

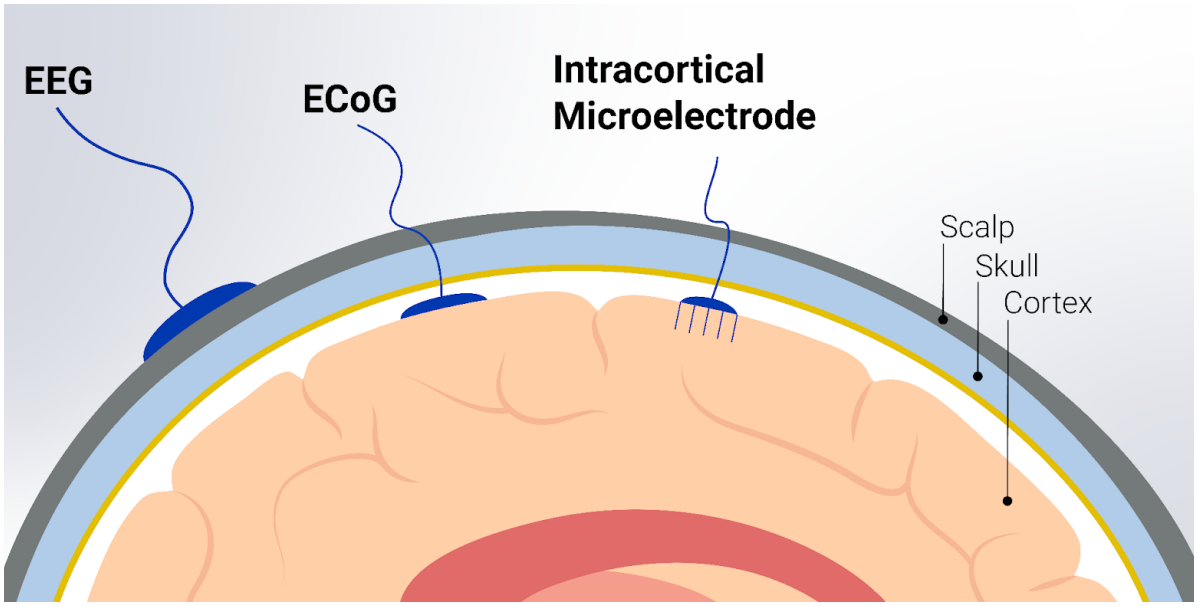

Figure 1 - Three types of prominent BCI sensorsBCI is divided into three distinct branches based on sensor placement: EEG, ECoG, and intracortical microarray. Copyright ©2022 Paradromics

4- Control mapping: The classified signals are subsequently transferred to external device or system control commands.

5- Device or system control: The control commands are used to control the external device or system.

1.2 Types of Brain-Computer Interfaces

BCIs may be split into two categories based on the placement of sensors on the brain: invasive and non-invasive. Noninvasive approaches employ sensors that contact the scalp, such as electroencephalogram (EEG), without any surgery. Conversely, invasive approaches require intracranial surgery to implant electrodes or signal receptors. As shown on Figure 1, Electrocorticography (ECoG) and intracortical neural recordings using microelectrode arrays (MEA) are two of the most prominent invasive recording technologies. EEG is one of the oldest and most widely utilized methods for examining the electrical activity of the brain to control external devices or systems. In an EEG-based BCI system, EEG signals are recorded via electrodes

placed on the scalp. Different states of brain function are associated with the various frequencies of EEG activity. Principal frequency ranges include:

1. Delta (0.5-4 Hz) – associated with deep sleep.
2. Theta (4-8 Hz) – associated with drowsiness, relaxation, and light sleep.
3. Alpha (8-13 Hz) – associated with wakeful relaxation and a calm, alert state.
4. Beta (13-30 Hz) – associated with wakeful, active, focused attention.
5. Gamma (30-100 Hz) - associated with cognitive processing and perception.

It is important to note that the exact frequency ranges for each group may change depending on the electrode placement and that the categorization of brain signals is not entirely clear-cut. Typically, EEG-based BCI applications are restricted to continuous movement control with a limited degree of freedom and discrete selections [14]. EEG sensors are commonly used to measure Field Potentials (FPs) to control computer cursors [15–18], spelling devices [19], assistive devices [20], hand orthoses [21], functional electrical stimulation (FES) of a patient muscle [22], robotic and prosthetic devices [23,24], and wheelchairs [25,26].

Electrocorticography, or ECoG, is a technique for measuring the electrical activity of the cerebral cortex, the brain's outermost layer. ECoG electrodes, which are thin metal discs or strips as shown in Figure 2, are surgically implanted onto the surface of the brain. The two primary forms of ECoG are epidural and subdural. Epidural ECoG involves the placement of electrodes on the surface of

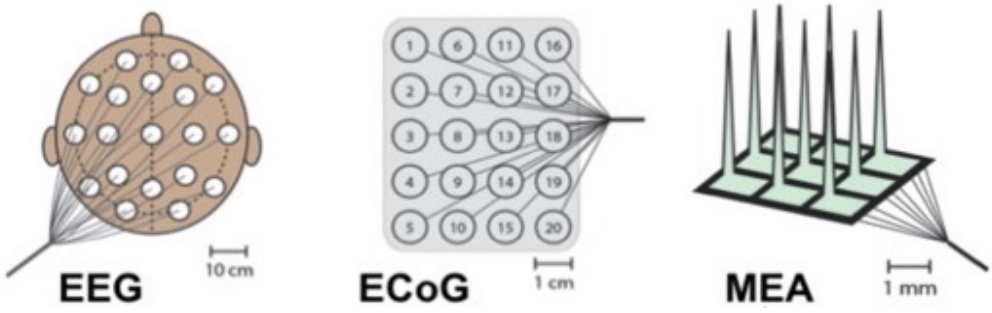

Figure 2 - Sensors of three different BCI technique

the dura mater, which is the outermost membrane covering the brain. This form of ECoG is less invasive than subdural ECoG and is often used for shorter-term monitoring or on individuals who are ineligible for subdural ECoG. However, the signal quality and spatial resolution[1] of epidural ECoG are often inferior to those of a subdural ECoG (the spatial resolution for epidural ECoG is about 1 cm and 0.5–3 mm for subdural). In subdural ECoG, electrodes are placed directly on the surface of the brain, beneath the dura mater. ECoG has significant advantages over other, lower-resolution techniques for measuring brain activity, such as EEG. Since ECoG electrodes are in direct or very close contact with the brain when measuring neural activities, they offer a greater spatial resolution, a lower noise level, and a higher frequency range (about 0.5 Hz to 300 Hz) than the EEG [27]. This enables more accurate localization and analysis of neural activity in deeper brain areas than the EEG. ECoG, on the other hand, is less invasive than intracortical neural recordings.

While the earliest known use of ECoG recordings on a human patient involved epidural electrodes, subdural electrodes have become the standard for intraoperative epilepsy monitoring in the twenty-first century. Epidural ECoG has gained popularity as a high-resolution alternative to invasive intracortical recordings for brain control of neuro-prosthetic devices, and it has been the focus of next-generation closed-loop devices capable of modulating therapies based on real-time cortical recordings [28–30].

Microelectrode arrays (MEAs), also known as the Utah Array, are implantable arrays of electrodes used to record the electrical activity of neurons within the cortex. They are usually made of thin wires or strips of conductive material and can be as small as a few millimeters across. MEAs can be used to record the activity of individual neurons called action potentials (APs) or groups of

[1] Spatial Resolution refers to the size of the smallest feature that can be detected by a sensor.

neurons called local field potentials (LFPs) and can be used to stimulate the brain as well as recording from it. Individual neurons generate APs, also referred to as spikes, when they receive sufficient stimulus from pre-synaptic neurons. APs are rapid (on the scale of milliseconds) and large (on the order of millivolts) fluctuations in the neuron's membrane voltage. They allow neurons to communicate with one another and transmit information throughout the brain. Recording APs from individual neurons requires small electrodes (micron scale) placed in very close proximity. Local field potentials LFPs are large-scale electrical signals generated by the activity of a particular population of neurons in a particular region of the brain.

In general, MEAs are implanted by making a small incision in the scalp and drilling a hole through the skull over the desired location in the brain. The MEA is then inserted into the brain through the hole and positioned so that the electrodes are in contact with the brain tissue. The MEA may be secured in place with a small screw or anchor, or it may be held in place by the surrounding brain tissue. Once the MEA is in place, it is connected to a wire or cable that runs through the skull and out of the scalp. This wire or cable is then connected to a recording or stimulation device, which can be used to monitor the activity of the neurons being recorded by the MEA or to stimulate specific brain regions.

Importantly, the electrodes in an MEA are not physically placed on top of specific brain cells (neurons). Instead, they are placed close to the cells, generally within a few hundred micrometers, and can detect the electrical activity of the neurons as they communicate. This activity is transmitted through the electrodes and out of the brain via the wire or cable, where it can be recorded or analyzed. MEAs are able to record individual neuron action potentials, unlike the previous two technologies, which could only record the average activity of multiple neurons. With the ability to directly monitor action potentials, the MEA allows for more precise measurements

of brain signals and the ability to target specific brain regions than the EEG and ECoG. MEAs can record higher frequency ranges than EEG and ECoG due to the fact that brain tissue, scalp, and brain skin act as a low-pass filter in EEG and ECoG, filtering the higher frequencies of neural activity. This high spatial resolution of recording neural activity in a specific brain region provides more information-rich data for machine learning algorithms, theoretically enabling the system to reach a higher degree of freedom BCI. Recent advancements in MEAs and intracortical surgical techniques have reduced the risks associated with the implantation of MEAs, and due to the need for higher quality neural signals, the majority of research in these fields focuses on the application of MEA-based BCIs, which compels us to examine three different studies on this type of BCI.

2 High-performance brain-to-text communication via handwriting

2.1 Introduction

This study [31] investigates whether handwriting, which requires quick and highly dexterous motor skills, can be identified and decoded via BCI. The participant is a male right-handed, 65-year-old who sustained a spinal cord injury approximately 9 years before joining the study. Two 96-electrode intracortical arrays (NeuroPort™ arrays [32] with 1.5 mm electrode length, Blackrock Microsystems [33]) were implanted in the participant's dominant left-hemisphere precentral gyrus, also known as the "hand knob". Attempts at handwriting revealed only a few little movements of the participant's right hand, but he retained full control of his head, face, and shoulders.

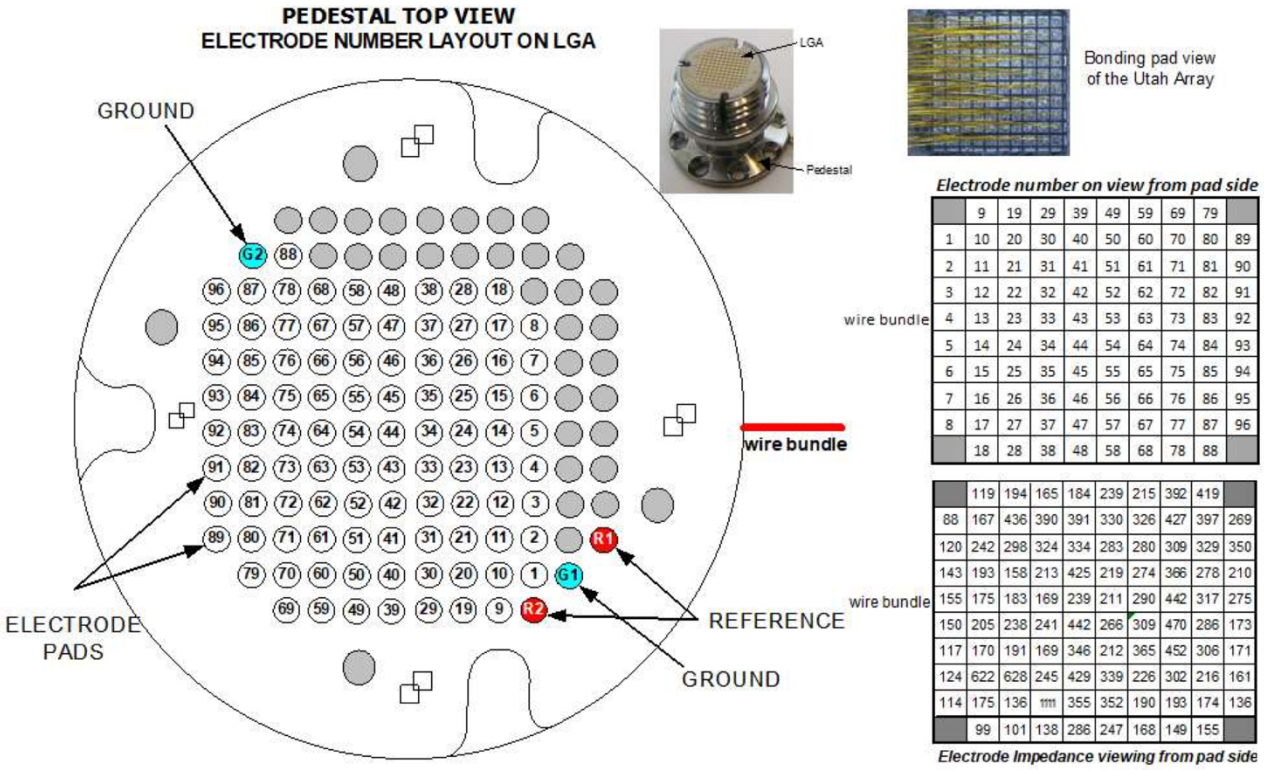

Figure 3 - The Black Rock MEA: On the left is the electrode pedestal pinout, on the right-top is the electrode array, and on the right-bottom is the corresponding resistance of each electrode.

The user is encouraged to "attempt" to write by imagining a pen on lined paper, and the system attempts to decode the writing through BCI. The decoding process may occur in online or offline mode, with offline mode providing greater precision.

As shown in Figure 3 each MEA comprises 100 land grid array (LGA) needles, of which 96 are signal detectors, two are connected to the ground, and two serve as reference points. The LGA side of the MEA is inserted into the gray matter cells on the outermost layers of the cerebrum, with the connector pins protruding from the scalp in a pedestal-like form. Through this connector, all signals are transmitted to a NeuroPort™ data aggregation system (Blackrock Microsystems).

2.2 Data Collection Method

This study includes two types of data gathering sessions. First, *copy typing sessions* (as depicted in Figure 4a) include data gathering blocks for “supervised single-letters” and “supervised complete sentences” that are primarily used to train machine learning algorithms. Following a 30 minute break period for algorithm training, evaluation blocks are used to evaluate the performance of the machine learning algorithms on held-out sentences from the training collection. The second type of data gathering are *free-typing data collection sessions* (Figure 4b), which begin with data collection of unsupervised phrases and unsupervised complete sentences from the participant's memory. After a break for machine learning training, the session concludes with free answer

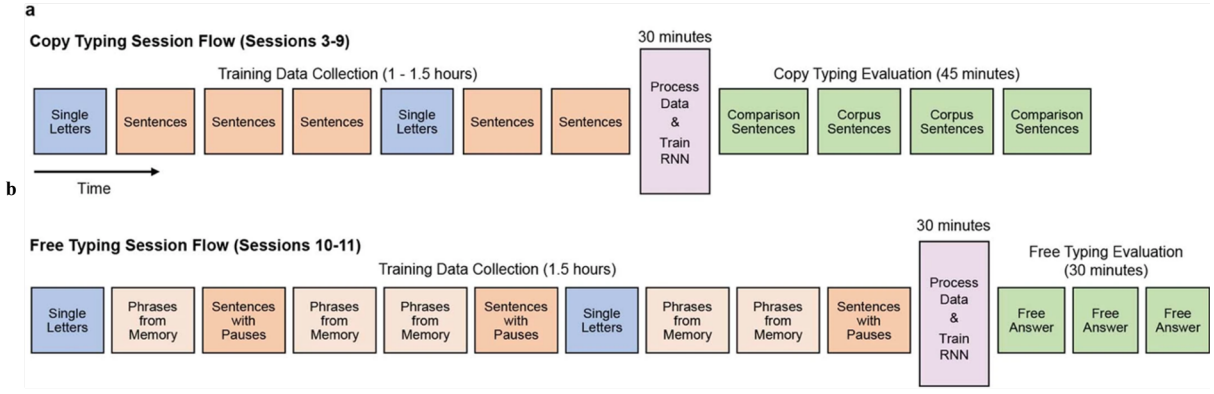

Figure 4 - The blocks of two types of data gathering sessions. Adapted from [31]

blocks, in which the individual is asked a random question and his response is decoded. There may be differences in neural activity of copying sentences from the screen and of recalling sentences or phrases from the memory; therefore, the sessions of free-typing data collection were meant to train machine learning algorithms for these differences. The participant always observes the decoded characters and indicates any decoder errors.

As depicted in Figure 5 for the supervised single-letter data collection, a monitor in front of the participant displays a single character. A red cube and a character appear on the screen for two to three seconds (the period duration was drawn from an exponential distribution with a mean of 2.5 seconds to imitate a realistic setting). The participant must read the character and get ready to write it when the cube turns green. The participant will have one second to write the character, and after that, one cycle of character data collection is concluded, and the next character and the red cube

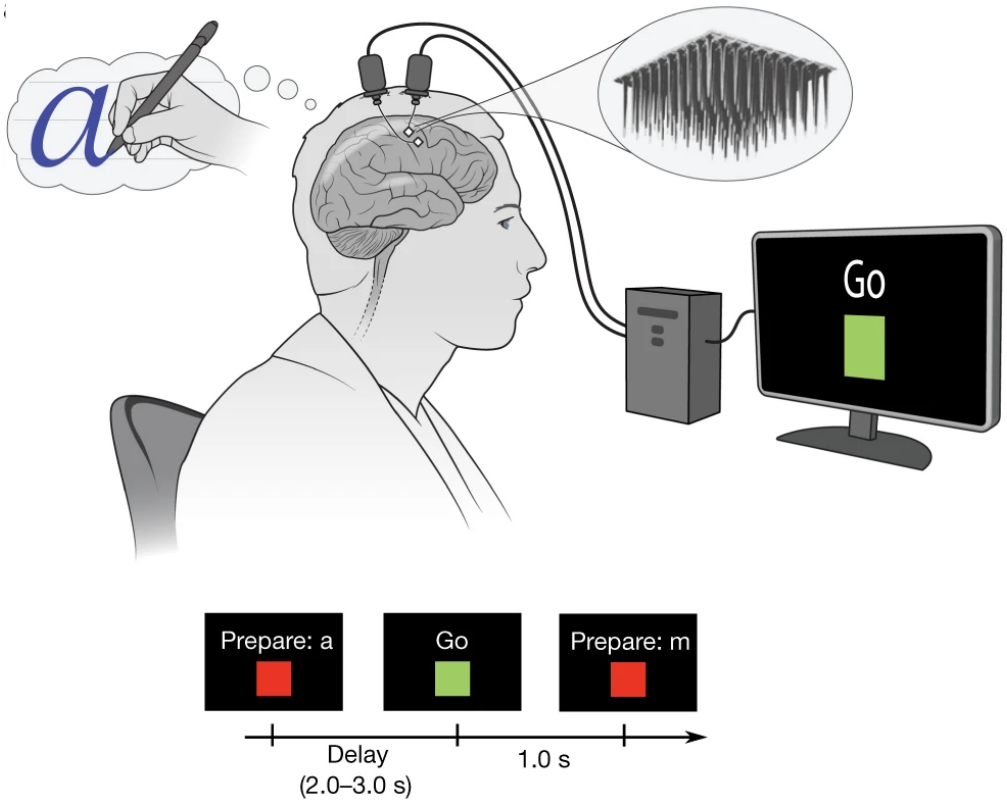

Figure 5 - The schematic of the system and process of supervised single-characters detection. Adapted from [31]

are displayed. Several 5-10 minute blocks containing a series of uninterrupted trials were used. Each of the chosen characters (a to z, a space, a period, an apostrophe, and a question mark) goes through 10 repetitions of training that is specific to that character in random order.

During the supervised sentence-gathering blocks (Figure 6), an entire sentence is shown to the participant. After allowing the participant sufficient time for acclimation, the subject is cued to begin writing the sentence. The participant can commence a new sentence by tilting his head to the right after completing the present one. This movement of head is optically recorded by the OptiTrack V120 Trio bar [34], which consists of three infrared cameras that track the location of markers worn on a headband. Each data recording session required the participant to compose between 84 and 102 sentences. With the unsupervised phrases or sentences, the primary difference is that the monitor displays merely a question, and the participant imagines writing a phrase or sentence from memory.

2.3 Signal Processing

Analog neural signals recorded by MEAs were bandpass filtered between 0.3 Hz and 7.5 kHz and then digitized with a resolution of 250 nV using a 30 kSps, 16-bits analog to digital converter

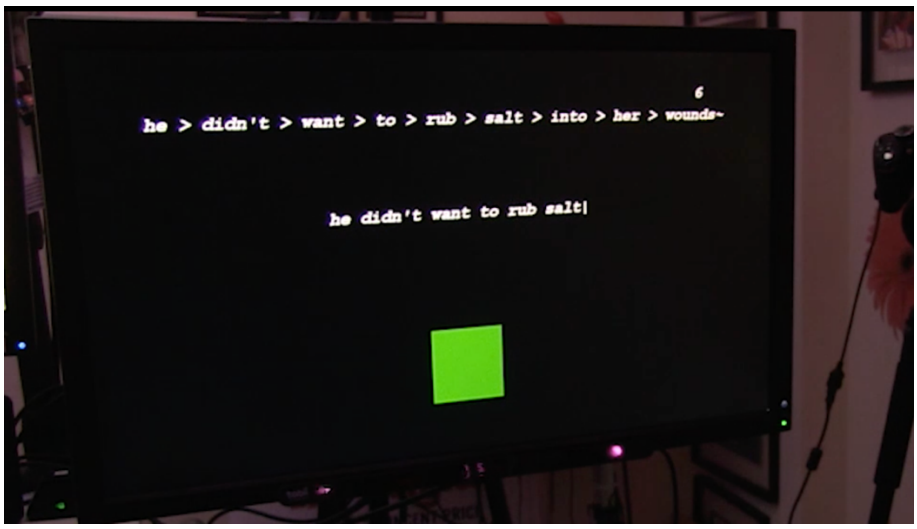

Figure 6 - The screen of supervised sentences detection

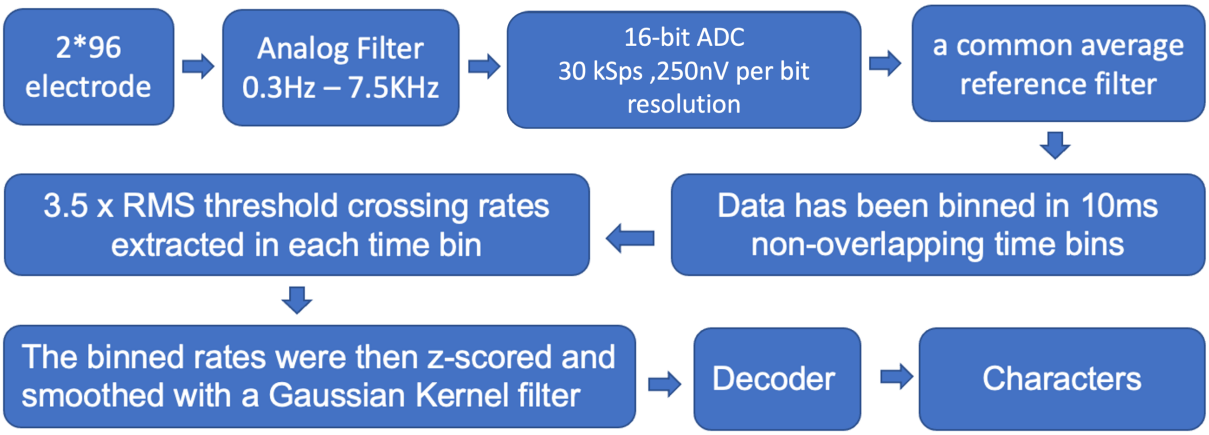

Figure 7- The block diagram of the system

(ADC). A common average reference filter was employed after digitization to decrease common mode noise by eliminating the average signal from each electrode in the array. Before threshold crossing detection, a digital bandpass filter from 250 to 3000 Hz was applied to each electrode. This is the frequency range where APs and LFPs occur. This filter was applied non-causally (with a 4ms delay) to enhance spike detection.

Sorting action potentials is computationally expensive and requires a high sampling rate. Alternative methods, such as thresholding, focus solely on the number of spikes that exceed a predetermined threshold level in a specific time bin. Various studies have been conducted to specifically describe the distinction between spike sorting and threshold crossing rates. Recent studies have shown that neural activities may be accurately distinguished from threshold crossing rates alone [35] and that neural decoding performance (within 5%) is comparable to that of employing sorted units [36–38]. Therefore, the rate of threshold crossing happens to be adequate for the decoder to identify the participant's handwriting attempts.

To accomplish this, the data from each channel was binned into non-overlapping time series of 10 ms for analysis and 20 ms for decoding. The rates of threshold crossing in each time bin were computed using a threshold of negative 3.5 times the root mean square (RMS) of the entire data

for that electrode. The rate for each time bin was determined by dividing the number of threshold crossings on that bin by the bin's duration. The binned rates were then z-scored by subtracting the average of each electrode and dividing by its standard deviation. Z-scoring helps to determine the distinctions between the firing of each electrode in idle mode and during the experiment. A Gaussian kernel with a standard deviation of 30 ms for single-character data and 40 ms for sentence data was then used to smooth this data. The process has been depicted in Figure 7.

2.4 Time Warping

Detecting each character's start and stop time is essential for predicting participant's intended phrases and sentences. The participant's writing pace is inconsistent, and the latency between the "go" cue and onset of his effort to write is unknown. Consequently, this research used a novel time-warping technique to estimate the start time of each character writing attempt and to make each character the same duration. The time-warped principal component analysis (TWPCA) Python library [39–41] was used to identify the two components of time warping, namely shifting across time, and stretch/compression. This package generates a predetermined number of principal component analyses (PCAs) (ten in this case) accompanied with regularized time-warping functions that align PCAs of each trial with the same character in offline mode. Figure 8b displays a PCA of time binned, z-scored and Gaussian Kernel smoothed threshold crossing data for 27 repetitions of a single character. Figure 8c shows the time warped version of the same data. In

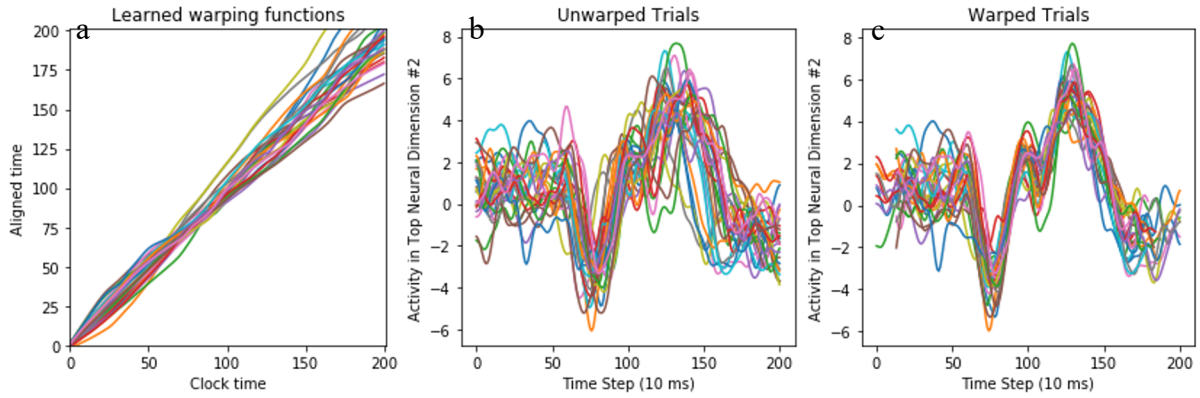

Figure 8 - TWPCA process for one character with 27 repetitions

extremes of the time warping are fixed, and each character has specific endpoints based on the normal amount of time required to write it.

2.5 Hidden Markov Model

Time warping cannot be used for real-time decoding because it is computationally expensive. Thus, to determine the start and end times of each character (hidden states) from the binned firing rates (observed states) without using the TWPCA, a Hidden Markov Model (HMM) is employed. This model identifies the times in which the participant imagines writing the character on a paper. The HMM model categorizes other hand movements and hand stillness as idle time. This model just handles the start and end of active writing times and does not classify the characters. Since the TWPCA has distinguished the start and stop times of the "single-letter" data, it can be used as a training data for this HMM. Later, this HMM distinguishes the start and stop times of the "supervised complete sentences" from the binned firing rates. This technique, known as "forced alignment," has been applied to speech recognition [42]. The HMM, with states and state transition

Table 1 - Transition probabilities for the HMM Model

States	Description	Transition Probabilities
$s_{x,j}$ for x<N-1	All states before the second to last, for character *j*	P ($s_{x,j} \rightarrow s_{x,j}$) = 0.2 P ($s_{x,j} \rightarrow s_{x+1,j}$) = 0.6 P ($s_{x,j} \rightarrow s_{x+2,j}$) = 0.2
$s_{N-1,\,j}$	Second to last state for character *j*	P ($s_{N-1,\,j} \rightarrow s_{N-1,j}$) = 0.2 P ($s_{N-1,\,j} \rightarrow s_{N,j}$) = 0.8
$s_{N,\,j}$	Last state for character *j*	P ($s_{N,\,j} \rightarrow s_{N,\,j}$) = 0.2 P ($s_{N,\,j} \rightarrow B_j$) = 0.1 P ($s_{N,\,j} \rightarrow s_{1,j+1}$) = 0.7
B_j	Blank state for character *j*	P ($B_j \rightarrow B_j$) = 0.5 P ($B_j \rightarrow s_{1,j+1}$) = 0.5
$s_{N,\,M}$	Last character state in the sentence	P ($s_{N,\,M} \rightarrow s_{N,\,M}$) = 0.7 P ($s_{N,\,M} \rightarrow B_M$) = 0.3
B_M	Last blank state in the sentence	P ($B_M \rightarrow B_M$) = 1.0

probabilities as shown in Table 1, defines an ordered march over the characters of each sentence in the training data.

The HMM model's states are represented as $S_{i,j}$, where j stands for the sentence's character number and i iterates through the states within each character. N is the number of states (time bins) in character j, which is different for each character. Some characters, like "l" are simpler to write and so require fewer time bins. Other characters, like "m", are more difficult to write and thus require more time bins. M is the total number of characters for that sentence.

2.6 HMM Viterbi

Once the system knows when each character begins and ends, it is easier to identify the character itself. To distinguish the character from the series of time bin data, a Viterbi algorithm was implemented (Figure 9). To reduce the number of iterations for each character, the probable time of occurrence for each character was calculated based on a table containing rough writing time estimates for each character, and the Viterbi algorithm was only iterated 0.3 T before and after this probable time, where T is the total time for each sentence. This constrains the HMM detector based on known character templates generated by time-warping.

It is possible that the speed of writing each character within the sentence data differ from the warped data of the individual characters. In order to refine the HMM emission probabilities and

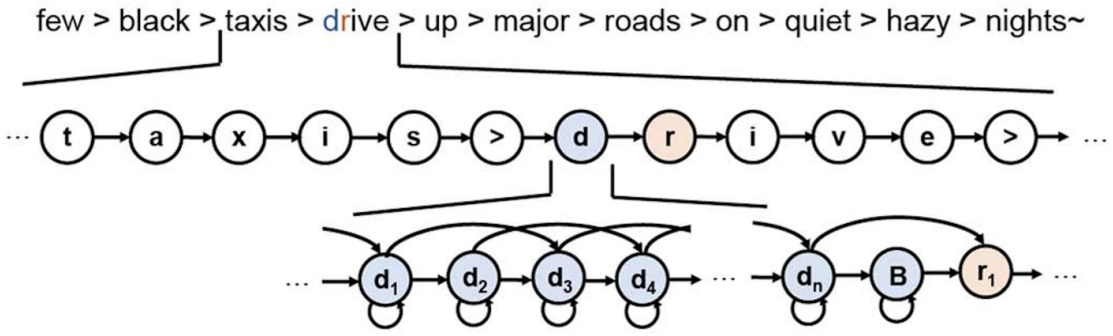

Figure 9 - The HMM Viterbi process for time bins of neural data threshold crossing. Each d_i is a 10 ms time bin, B is the blank period. Adapted from [31]

the characters' start time based on these speed variations; a grid search was conducted using a stretch factor for the time bins of the neural activations to maximize the correlation of each character with the observed activity. The stretch factor has been calculated in 15 linearly spaced increments, ranging from 0.4 to 1.5, and illustrates how the character template gets stretched or shrunk over time to be longer or shorter than its average. The probabilities of HMM emissions have been adjusted based on this refinement, whereas the probabilities of transition have remained the same. With the updated HMM emission probabilities, the Viterbi algorithm has been executed once more to make the results more precise. These character start and stop times and the character detections are used to train a recursive neural network (RNN).

2.7 Synthetic Data Generation

The amount of data available for this experiment may not be sufficient to train complicated algorithms such as RNNs. A small sample size may result in the model being overfit. This study addressed this limitation by producing a new collection of sentence data using the labeled data from the preceding section. To do this, binned spikes, from the start of one character to the start of the next, were taken from the database and used to generate new sentences. This ensures the inclusion of pauses and transition-related activities that are plausible in real-world data, necessitating their inclusion in synthetic data. Then, with binned spikes for each character and a predetermined set of 10,000 popular English words selected at random [43], a new collection of sentences were constructed. The neurological data of handwriting is substantially connected with the pen trajectories, suggesting that the start and stop time bins of each character may be highly

Table 2 - Starting point of characters divided to four groups.

Start Height	0	0.25	0.5	1
Character	comma	a, o, e, g, q	c, d, m, j, i, n, p, r, s, u, v, w, x, y, z, space (>), period	b, t, f, h, k, l, apostrophe, question mark

correlated with the character before or after it, depending on their start and end positions. For example, the letter 'b' begins at highest point and ends at the lowest point. If the next character is a 'c' that begins in the middle, the brain activity in time bins near to the transition point will be different than if the next character is an 'l' that begins at the highest point. In order to address this issue, characters have been separated into four groups based on their starting position (see Table 2). These category discrepancies have been used to generate synthetic data that is as close as feasible to the actual data. A random selection was made from among all data samples in the library whose following character in the training data started at the same height as the following character in the synthetic sentence.

2.8 Real Time Decoder

The neural activities are temporally binned, z-scored, and smoothed using a Gaussian kernel denoted by x_t in Figure 10. Later, a battery of artificial noise was added to these signals to prevent the decoder from over fitting. Details of these artificial noises and their effects are discussed in section 2.12. A linear affine filter was then implemented to account for daily fluctuations of neural activity as below [44–46].

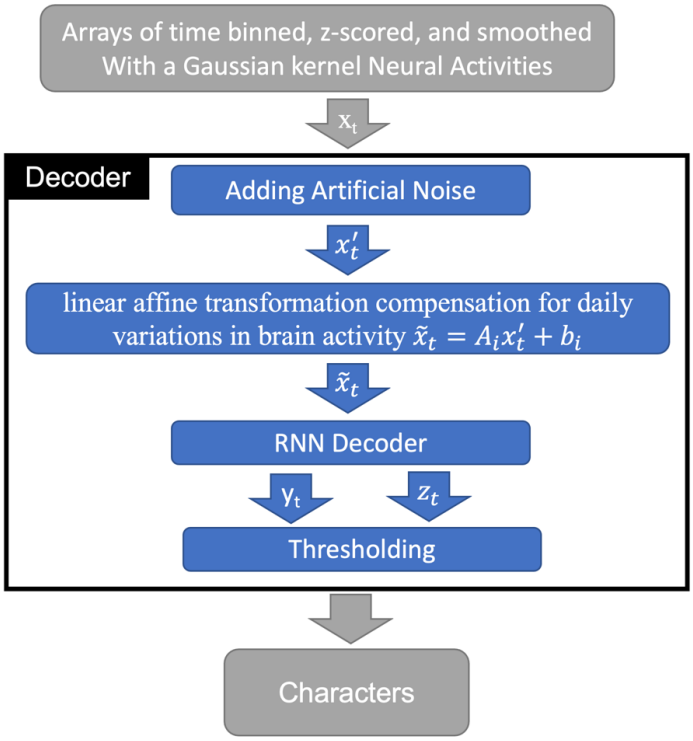

Figure 10 - The block diagram of the Real Time Decoder

$$\tilde{x}_t = A_i x'_t + b_i \qquad (1)$$

Where:

x'_t is a 192-dimensional vector of neural features (one dimension for each electrode) at every time t, with artificial noise introduced.

A_i is a 192 x 192 coefficient matrix.

b_i is a 192 x 1 coefficient vector.

These signals are passed to a RNN to determine the probability of each of the 31 characters occurring (y_t) and the likelihood of a new character occurring (z_t). The characters are decoded based on these probabilities.

2.9 RNN Architecture

The neural activity was transformed into a time series of character probabilities by a gated recurrent neural network (GRU)-RNN as shown in Figure 11. A GRU is a type of RNN that uses gates to control the flow of information in the network. Like traditional RNNs, a GRU processes sequential inputs one step at a time, but it uses gates to control the flow of information between time steps. This allows the network to better preserve information from earlier time steps and prevent the vanishing gradient problem that can occur in traditional RNNs. The vanishing gradient

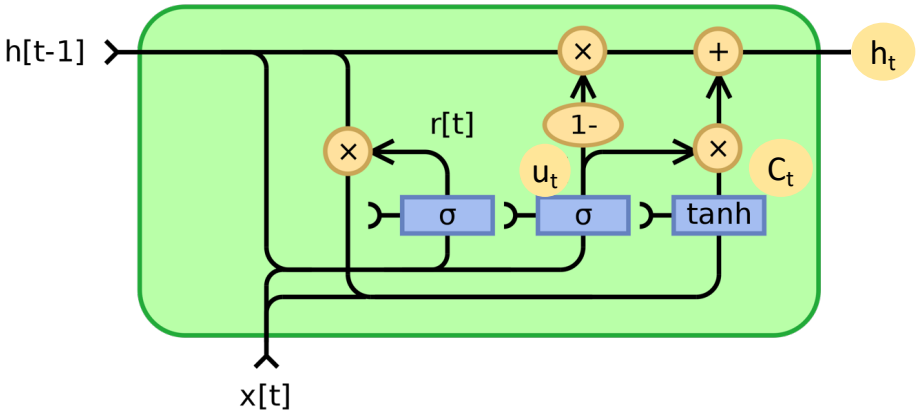

Figure 11 - Architecture of GRU. Copyright @2020 paperswithcode.com

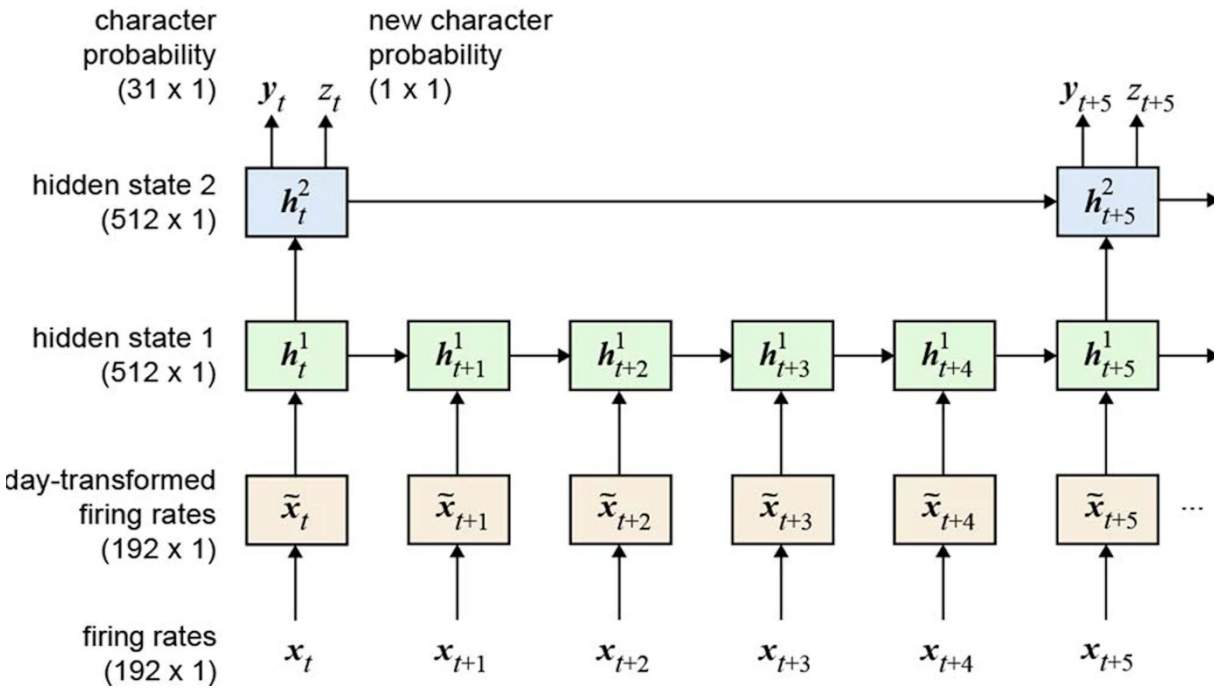

Figure 12 - The RNN architecture. Adapted from [31]

problem is a difficulty in training recurrent neural networks (RNNs) that occurs when the gradient of the loss function regarding the network's weights becomes very small, making it challenging to update the weights using backpropagation. This issue can arise when the RNN is attempting to discover long-term dependencies, as the gradient is multiplied by the network's weights as it propagates back through the layers. Consequently, the gradient can become very small by the time it reaches the network's early layers, making it difficult to update the weights in these layers. In this study, a variation of the conventional GRU was employed. This GRU's gating is determined by the following formulas.

$$\begin{aligned} \mathrm{r_t} &= \sigma(\mathrm{W_r}\tilde{x}_t + \mathrm{R_r h_{t\_1}} + \mathrm{b_{Wr}} + \mathrm{b_{Rr}}) \\ \mathrm{u_t} &= \sigma(\mathrm{W_u}\tilde{x}_t + \mathrm{R_u h_{t\_1}} + \mathrm{b_{Wu}} + \mathrm{b_{Ru}}) \\ c_t &= \sigma_h(W_h\tilde{x}_t + r_t * (R_h h_{t\_1} + b_{Rh}) + b_{Wh}) \\ \mathrm{h_t} &= (1 - \mathrm{u_t}) * \mathrm{c_t} + \mathrm{u_t} * \mathrm{h_{t\_1}} \end{aligned} \qquad (2)$$

Where:

$\tilde{x}_t$ is the input vector at time step t.

h_t is the hidden state vector at time t.

σ is the logistic sigmoid function.

σ_h is the hyperbolic tangent.

r_t is the reset gate vector at time t.

u_t is the update gate vector at time t.

c_t is the candidate hidden state vector at time t.

W and R are weight matrices.

b is the bias vector.

* Denotes element-wise multiplication.

The input for the second layer (blue blocks on Figure 12) is the h_t of the first layer (green blocks), which is indicated as $h^1{}_t$. First layer runs once every 20 ms and second layer runs once every 100 ms, resulting in a 100 ms delay for the decoder. The GRU-RNN was trained with a significant output latency, anticipating character probabilities from one second prior; this was required to ensure that the GRU-RNN had sufficient time to process the complete character before determining its identity. The following formula was used to derive the output probabilities from the hidden state of the second layer:

$$y_t = \mathrm{softmax}\left(W_y h_t{}^2 + b_y\right) \quad (3)$$

$$z_t = \sigma(W_z h_t{}^2 + b_z)$$

σ is the logistic sigmoid function.

$h_t{}^2$ is the hidden state of the second layer.

W and *b* are weight matrices and bias vectors.

y_t is a vector of character probabilities (one entry for each character).

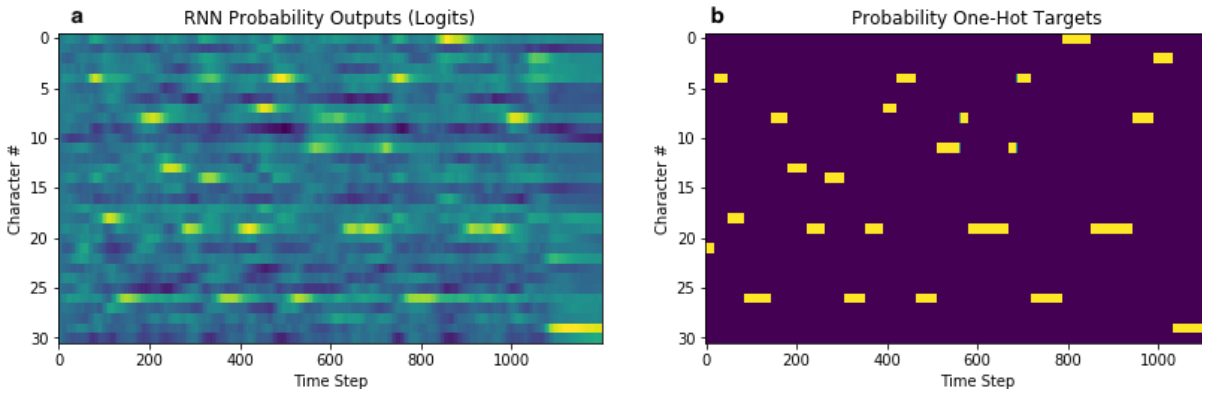

Figure 13 - The output of the GRU-RNN

z_t is a scalar probability that represents the probability of any new character beginning at that time step.

Real-time operation involves thresholding the z_t (threshold = 0.3) to determine when to emit a new character. If z_t crosses the threshold, the most likely character is emitted 300 ms later in y_t. In Figure 13a, the probabilities of each character at various time steps are represented, and in Figure 13b, the threshold values for releasing one-hot character at a time are shown.

2.10 Comparison of GRU-RNN to an HMM decoder

To evaluate whether an GRU-RNN was required for high performance, the performance of a simple HMM decoder was investigated. According to the findings, the GRU-RNN outperforms a basic HMM, especially in held-out blocks where neural non-stationarity is likely to have resulted in considerable changes in feature means [47,48]. Nonetheless, when feature mean drift is taken into consideration, even an HMM decoder can perform reasonably well, indicating that brain activity is highly distinguishable. As with the forced alignment HMM decoders that were used to label the sentence data, the HMM decoder also contains character states that can transition to any other character with equal probability instead of marching forward through a fixed sequence of characters. Evaluating the GRU-RNN and HMM models without a language model cannot be a

Table 3 - Error rates of the HMM and the GRU-RNN in this study

Decoder	Train + Mean Subtraction	Train	Test
GRU-RNN	0.23 %	0.23 %	0.70 %
HMM	2.96 %	6.70 %	80.08 %

fair comparison since the GRU-RNN itself could learn character transition probabilities that the HMM cannot. A comparison of the character error rates in offline performance between the GRU-RNN decoder and the HMM decoder is presented on Table 3.

Based on the results, GRU-RNN outperforms HMM, significantly when the feature means have changed over time. As the HMM lacks a mechanism for adapting to changes in the feature means (drifts over time in baseline firing rates), it performs poorly in generalizing to blocks that are held out and performs best when subtracting within-block feature means to account for any changes that have occurred.

2.11 Decoder retraining analysis

Originally, 50 calibrations were used for each session. It is interesting to calculate the performance of the decoder using fewer calibration data for copy-typing sessions. This was accomplished by running an offline GRU-RNN for copy typing sessions with fewer than 50 calibration sentences. The results are shown in Figure 14a, which quantifies the effect calibration sentences in reducing character error rates. A second question is how the duration of time since the last calibration day affects the character error rate Figure 14b shows the effect of time since GRU-RNN training on character error rates. Increasing the time since GRU-RNN training increases the character error rate significantly.

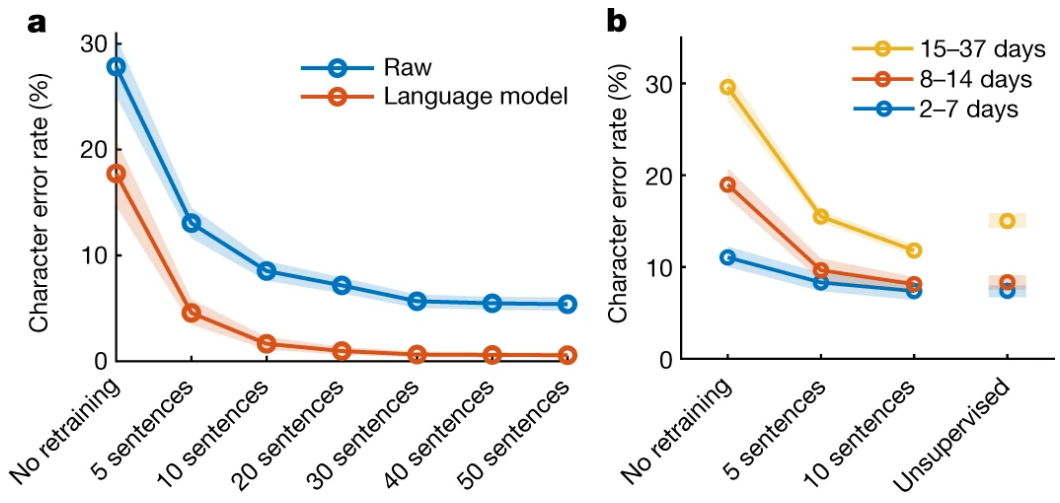

Figure 14 - The character error rate for the GRU-RNN : a) when trained with different number of sentences with and without the language model. b) for the GRU-RNN as a function of number of training sentences. Each trace refers to period between experimental sessions. Adapted from [31]

2.12 Adding Noise

Two types of artificial noise were added to the system to prevent the GRU-RNN from being overfit. As demonstrated in Figure 15a, adding white noise directly to the input feature vectors significantly increased performance. The addition of white noise to the inputs forces the GRU-RNN to map clusters of similar inputs to the same output, hence enhancing generalization.

In the meantime, the GRU-RNN has been made resistant to non-stationarity in neural data by introducing artificial alterations to the neural features' means. Intracortical BCIs have been hindered by the accumulation of drifts in baseline firing rates over time [49–51]. As indicated in Figure 15b the addition of artificial mean changes significantly enhanced the GRU-RNN's capacity to generalize to held out (test) data blocks. Constant offset noise and random walk noise were added to the system to simulate changes to the neural feature means. As indicated by equation (4) the three forms of artificial noises were combined to generate the vector input [54–56].

$$x_t' = x_t + \epsilon_t + \varphi + \sum_{t=0}^{t} v_i \quad (4)$$

x_t are the original neural features.

ϵ_t is a white noise vector unique to each time step.

φ is a constant offset vector.

v_i are white noise vectors that are cumulatively summed to simulate a random walk noise.

2.13 Language model

In a retrospective offline analysis, a custom, large vocabulary language model was employed to automatically fix decoder mistakes. The language model consisted of two stages: (1) a 50,000-word bigram model that processes the neural decoder's output to build a collection of candidate phrases, and (2) a neural network to rescore these candidate sentences [52,53]. Since all language models employ a complete list of characters, but the decoder is missing parts of them such as capital letters, hyphens, and newline, the language model must be adjusted and trained for this system.

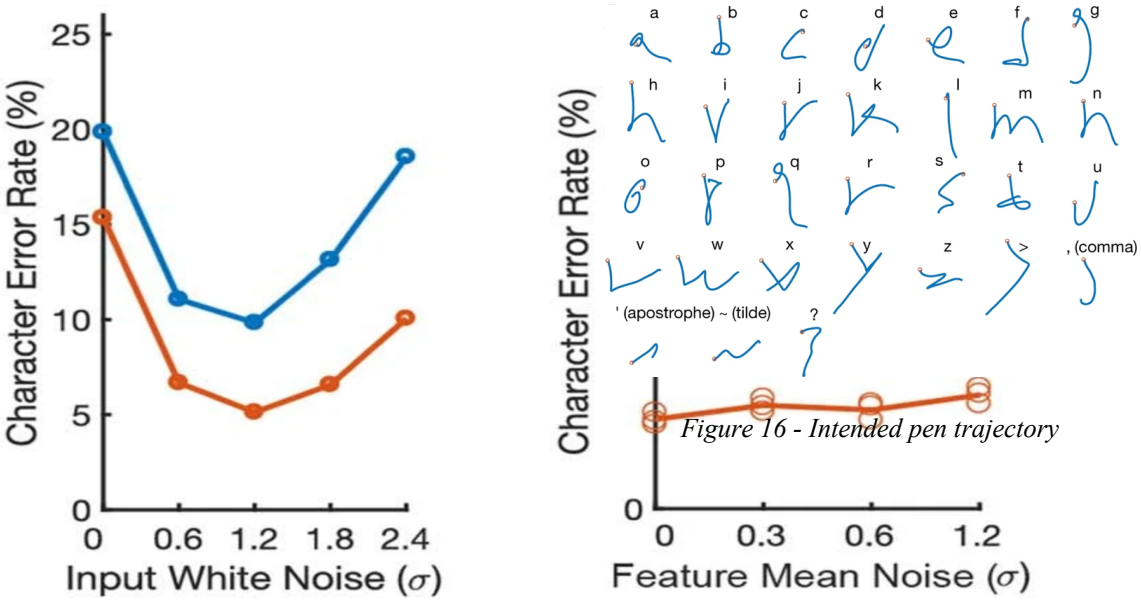

Figure 15 - Character error rate. in presence of artificial white noise(a) and artificial mean noise(b) Adapted from [31]

Figure 16 - Intended pen trajectory

The language model is essentially an HMM, with each hidden state representing a guess for the next letter and the state transition probabilities encoding the probabilities for which words are likely to follow others. The language model makes inferences by employing an approximation of the Viterbi algorithm to locate likely sequences of characters. The Viterbi search combines information from the language priors (state transition probabilities) and the neural decoder about which characters are expected to occur at any moment in time (observations).

2.14 Unsupervised Training

Before evaluating the GRU-RNN's performance in real-time, an unsupervised training survey was conducted only for the offline analysis utilizing all 50 words of training data acquired at the beginning of each day. This was intended to imitate the effect of running an unsupervised training method "in the background" that updates the decoder while the user types the normal sentences (and does not require prior knowledge of the characters in each sentence). The results are depicted in Figure 14b.

2.15 Pen trajectory visualization

A pen trajectory function based on neural activity has been developed solely for visualization purposes. The participant began by describing his handwriting, after which another individual attempted to replicate it physically on the screen using a mouse. Then, these simulated template writings were utilized to teach a system to visualize the individual's intended writing.

$$v_t = Dx_t + b \qquad (5)$$

v_t is a 2 x 1 vector holding the X and Y velocity of the pen tip at time t, D is a 2 x 192 decoding matrix, x_t is a 192 x 1 vector of binned threshold crossing rates, and b is a 2 x 1 offset term. Importantly, the decoding was cross validated by leaving out one character at a time. Thus, the pen tip velocities for any particular character were determined using a decoder trained on all previous characters, preventing the decoder from overfitting to high-dimensional neural input. The aforementioned templates then specified the target velocity vector for the decoder on each time step of each trial, similar to how previous research [54–56] has trained decoders to anticipate the user's "intended" velocity for continuous movement tasks. These templates were only meant to be a rough approximation of T5's intended pen tip velocities, based on the premise that an individual drawing the same character form with a computer mouse would naturally follow a similar velocity trajectory. The rough approximations are plotted on Figure 16.

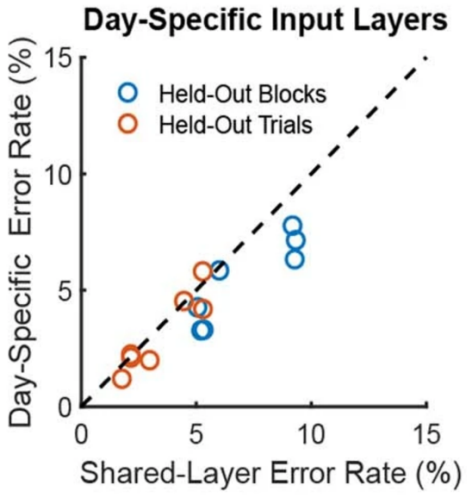

Figure 17 - Error rate comparison between using a shared and day specific input layer. Adapted from [31]

Table 4 - Summary of the performance. Adapted from [31]

	Without Language Model	With Language Model (Bigram LM)	With Language Model (Bigram LM + GPT-2)
Test Character error rate	5.32 %	1.69 %	0.90 %
Test Word error rate	23.28 %	6.10 %	3.21 %
Learn Character error rate	2.78 %	0.80 %	0.34 %
Learn Word error rate	12.88 %	3.64 %	1.97 %

2.16 Performance

The x-axis in the Figure 17 depicts the error rate while using a shared A_i and b_i for all days, but the y-axis depicts the error rate when using a day-specific A_i and b_i, demonstrating that using the day-specific input layer reduces the error rate in most cases. The summary of the performance is depicted in Table 4. Reaching rates of 90 characters per minute with greater than 94% raw accuracy online and greater than 99% accuracy offline with a general-purpose autocorrect indicates that this technology can be a highly important tool for integrating paralyzed individuals into society. Even years after paralysis, the brain representation of handwriting in the motor cortex is presumably robust enough to be helpful for a BCI, according to these findings.

3 A brain-computer interface that evokes tactile sensations improves robotic arm control

Tactile feedback is crucial for enabling BCI-assisted individuals with UE neurological disorders to perform ADLs independently, as it conveys state transitions such as object contact. Nevertheless, prosthesis users primarily rely on visual feedback. This study [57] demonstrates that intracortical stimulation of somatosensory cortex improves functional item transfer activities with a BCI-controlled prosthetic limb in a tetraplegic individual.

3.1 The participant

The participant is a 28-year-old male with tetraplegia caused by a C5 motor and C6 sensory American Spinal Injury Association (ASIA) B spinal cord injury sustained 10 years prior to the device's implantation. The "C5 motor level" corresponds to the fifth cervical (C5) vertebra, which is located in the spinal cord's neck region. The C5 spinal segment is responsible for shoulder elevation (deltoid muscle) and elbow flexion (biceps muscle). However, there may be a partial or complete loss of function below this level, including the hand and wrist-moving muscles. The "C6 sensory level" corresponds to the sixth cervical (C6) vertebra, which resides in the neck region of the spinal cord. At the C6 sensory level, the thumb side of the forearm and the thumb, index, and middle digits are sensitive. "ASIA B" is a classification used to characterize the severity of spinal cord injuries on the ASIA impairment scale. ASIA B indicates some preservation of sensory function below the level of injury to the nervous system, but no preservation of motor function.

3.2 The Micro Electrode Arrays

Two Blackrock [33] MEAs (88 wired platinum electrodes (1.5 mm long) in a 10x10 array) were implanted in the motor cortex of the hand and arm to interpret movement, and two other Blackrock MEAs (32 wired electrodes each, 6x10 array, 1.5 mm long, sputtered iridium oxide) were

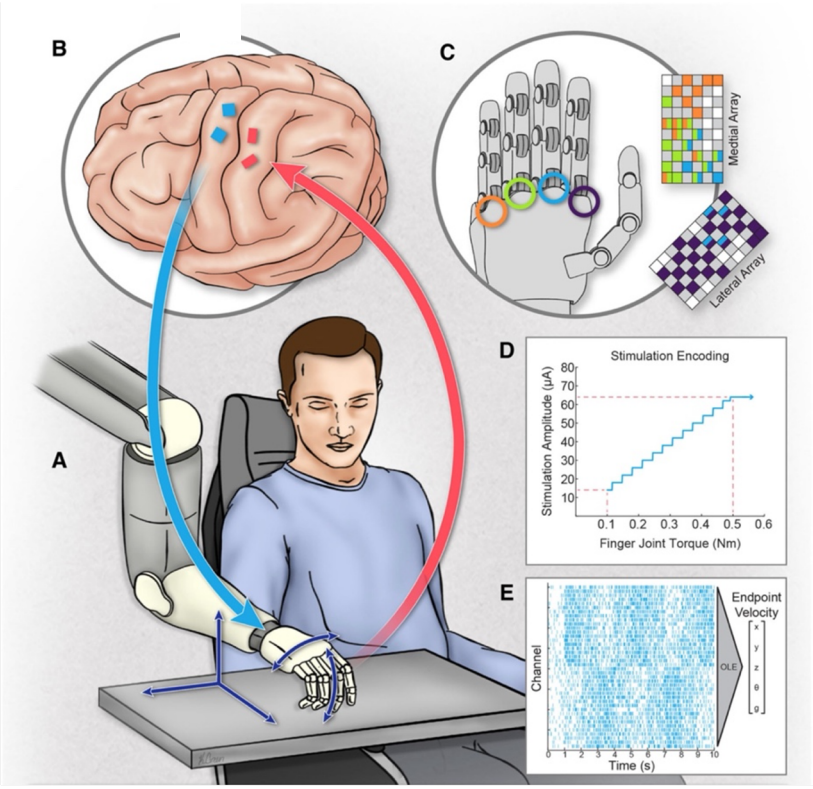

Figure 18 - A block diagram of the system and its various sections. a) The MPL robotic arm, b) The sensor placements on the brain, c) the effects of each electrode on the pressure feeling of the specific finger, d) The mapping between the current levels of the stimulation and the recorded torque and e) The mapping of the signals recorded from the brain with MEAs to the actual directions. Adapted from [57]

implanted in area one of the somatosensory cortex to trigger feelings in the right hand fingers by intracortical micro stimulation (ICMS) as depicted in Figure 18b. Neural voltage recordings from each electrode in the motor cortex were bandpass filtered between 0.3 Hz and 7.5 kHz and digitized at 30 kSps using a NeuroPort signal processor, and microsimulation-induced electrical artifacts were removed using a combination of digital signal blanking and filtering. During each stimulus pulse, a sample-and-hold circuit blanked the recorded signals. The signals were then high-pass filtered with a 750 Hz, first-order Butterworth filter that decreased the effect of additional transient discontinuities in the signal, allowing for rapid baseline settling of the wideband signal. This signal's spike threshold was set to -4.5 times its RMS. The software rejected any transient threshold crossings that occurred in the sample immediately after the blanking period.

3.3 Prosthetic Limb

The Modular Prosthetic Limb (MPL) [58] is a bionic limb whose dexterity, weight, range of motion, and force generation are comparable to those of a human arm. This will assist the system and participant feel as if he is moving his own hand, without which the brain would not be able to easily coordinate with the speed, range of motion, or force. Its purpose is to restore full mobility and functionality to the upper extremities of amputee soldiers. This prosthetic limb was not wearable, so it was only used as a proof of concept in this study. The system conveys the MPL's torque sensors data to tactile feedback via the ICMS.

3.4 Performance Analysis

In this study, two distinct performance analysis tests were conducted. For 1D movements as depicted in Figure 19b, as an object transfer task, a person must grasp an object and transfer it from the left box to the right box without lifting it. The second performance evaluation is a modified variant of the Action Research Arm Test (ARAT) [59,60] in which the subject grasps an object from the left box, lifts it, and places it on platform in the right box. The 3D movement of the object is led by the fact that the platform is 6 cm higher than the left box as depicted in Figure 19a. Only ARAT results are addressed in this article.

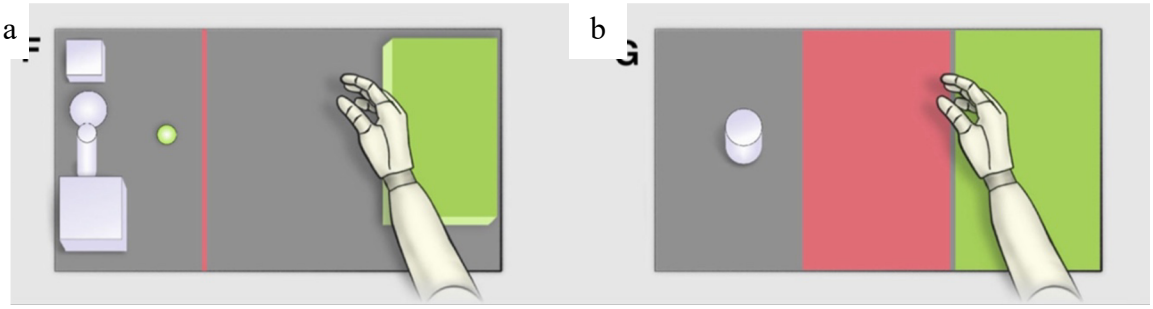

Figure 19 - Overhead view of ARAT (A) and Object Transform (B) tests. Adapted from[57]

3.5 Data Gathering

The trial was conducted 717 days following the implantation and spanned 27 days, with eight data collection sessions lasting four hours each. Over the course of these sessions, two feedback conditions were evaluated using a block-design: for the first four sessions, the bidirectional BCI with ICMS-evoked tactile feedback was driven by sensors on the MPL, and for the next four sessions, the same testing protocol was followed with the ICMS disabled. On each experiment day, a new decoder was trained without ICMS using observation and computer aid [61], and then the subject completed three blocks of the sequence task, five blocks of the object transfer task, and one ARAT session using brain control and no computer assistance.

3.6 Intracortical microsimulation

A biphasic, charge-balanced, current controlled pulse wave has been used for the ICMS. Using biphasic charge balance pulses will prevent injury to brain tissue resulting from unbalanced electrical charge injection. It indicates that the circuit controls the maximum current of each pulse while simultaneously stimulating one positive and one negative pulse with the same amount of electrical charge (the area under each pulse in a plot of current versus time). The cathodal phase lasted 200 μs, the anodal phase lasted 400 μs, and the amplitude of the anodal phase was tuned to half the amplitude of the cathodal phase to provide charge-balanced stimulation. The phases were separated by 100-second intervals resulting in a charge-balanced stimulation pulse wave of 1.25 kHz.

This type of stimulation pulse is chosen to maximize efficiency and minimize stimulation amplitudes which is essential for keeping the neurons healthy in long term. The stimulation signals must constantly be charge-balanced to ensure that there is no long-term polarization of the electrode-tissue interface, which would cause damage to the electrode materials and local tissue.

The theoretical justification for deploying a recovery (anodic) phase that is twice as long and half as intense (but has the same charge as the cathodic phase) is to deliberately slow down the recovery phase.

Detailed explanations of sensory perceptions induced by the ICMS of the somatosensory cortex have been reported previously [62,63]. This study identifies the area of the brain that, when stimulated, induces somatosensory activity at a certain body location. To provide somatosensory input during real-time prosthesis control, electrodes that could trigger perception centered on specific fingers were stimulated. One electrode with a projected field in the proximal interphalangeal joint of the index finger was mapped to the output of the torque sensor in the index finger metacarpal phalangeal joint of the MPL, as depicted in Figure 18c and Figure 20a. The torque sensor output from the middle finger of the MPL was mapped to four electrodes with projected fields in either the middle, ring, or little finger. The projected fields from the specified electrodes collectively included the index, middle, ring, and pinky fingers. MPL finger motor torques were linearly mapped to ICMS current amplitudes, such that increasing grab force increased the ICMS current amplitude and, consequently, the perceived stimulus intensity. Current stimulus amplitude is determined by:

$$A_t = \left(\frac{\tau_t - \tau_{min}}{\tau_{max} - \tau_{min}}\right) * (A_{max} - A_{min}) + A_{min} \qquad (6)$$

where A_t is the commanded pulse train current amplitude at time step t, A_{min} and A_{max} are the electrode-specific range of stimulus amplitudes, and τ is the torque sensor data that was utilized to transmit grip force. τ_{min} and τ_{max} are the minimum and maximum torque values corresponding to the minimum and maximum stimulation amplitudes, respectively. The specified torque levels were 0.1 Nm and 0.5 Nm, which roughly equate to a soft touch and a firm grip, respectively. These values were linearly translated to stimulus amplitudes ranging from 14 to 64 μA with 4 or 6 μA

increments (Figure 18d). The maximum and minimum currents are fixed for the participant. Another form of this formula can be written as:

$$\frac{A_t - A_{min}}{A_{max} - A_{min}} = \frac{\tau_t - \tau_{min}}{\tau_{max} - \tau_{min}} \quad (7)$$

According to this formula, a normalization measured torque is translated to a normalization stimulation current. Every 20 ms, new torque data was collected and utilized to adjust the pulse train's amplitude. This whole process, from sensor data collection to the generation of fresh stimulation pulses, happens within 20 ms, establishing a maximum delay between peripheral mechanical events and cortical stimulation.

a

b

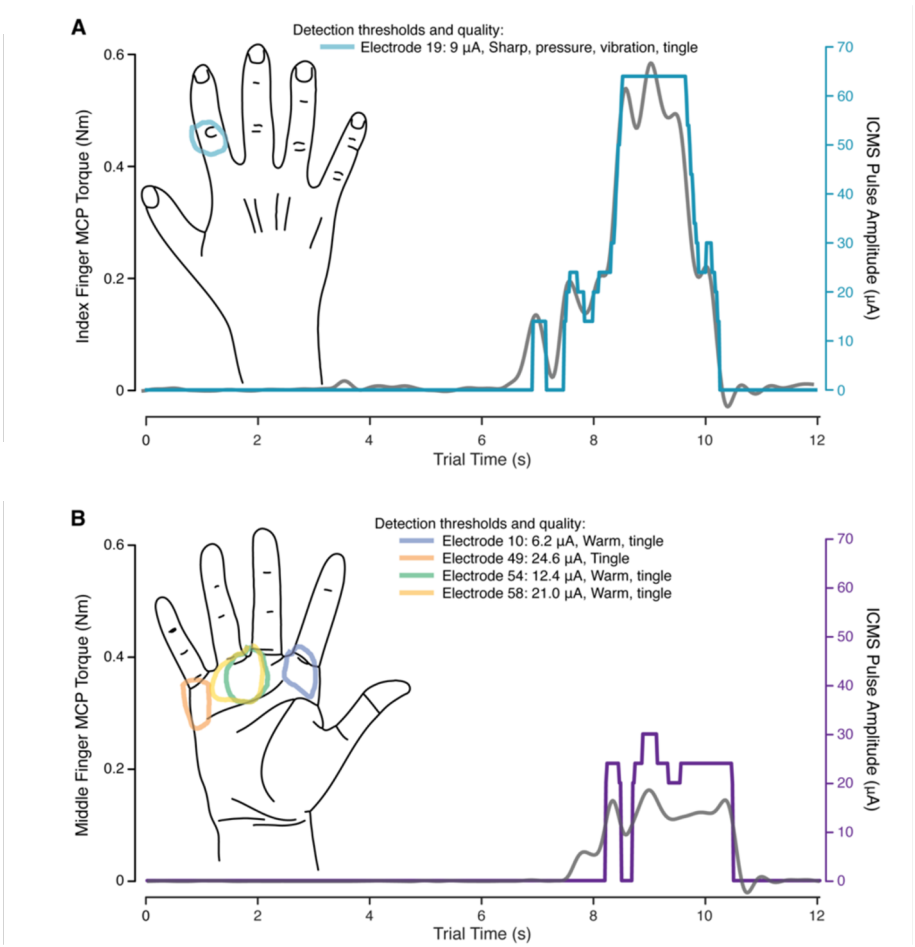

Figure 20 - Torque-to-force conversion example and electrode characteristics. Adapted from[57]

Figure 20 indicates a torque-to-force conversion example and electrode characteristics. An example of the torque and accompanying stimulation amplitude profiles obtained during an ARAT experiment using a 7.50 cm cube. In Figure 20a, the torque sensor data from the index finger motor regulated the stimulation amplitude on an electrode that stimulated the index finger proximal interphalangeal joint. The electrode's detection thresholds and perceptual quality are mentioned. On Figure 20b, the torque sensor data from the middle finger motor regulated the stimulation amplitude on four electrodes that elicited feelings at the bases of the second through fifth digits, as seen on the hand figure. Each electrode's detection thresholds and perceptual properties are listed. All electrodes were stimulated with the same amplitude.

3.7 Decoder Design

A decoder has been designed to regulate continuously and concurrently five degrees of freedom (DoF) of the endpoint velocity of MPL based on the threshold crossing rates of the neural activity of the brain. These five degrees of freedom include wrist pronation and supination as well as three-dimensional hand aperture, with the thumb always placed opposite the fingers. Using an inverse kinematic model and additional restrictions, the robotic arm's individual joint angles were controlled to maintain a realistic elbow position.

To train the decoder, the participant is shown a 3D virtual image of the MPL and instructed to envision moving it in specific directions until reaching a randomly determined location and grabbing a target. This activity also included an auditory cue to assist the person in completing it. Using an encoding model that relates neuronal firing rates to arm kinematics, an optimum linear estimator decoder was generated after monitoring the completion of 27 trials, which lasted around seven minutes. The model of encoding was:

$$f = b_0 + b_x v_x + b_y v_y + b_z v_z + b_\theta v_\theta + b_g v_g \qquad (8)$$

Where x, y, and z are translation dimension, θ is wrist rotation dimension, and g is grip dimensions, f is the square root transformed firing rate of a unit during movement described by 5 dimensional velocity vector $\mathbf{V}(v_x,v_y,v_z,v_\theta,v_g)$ and b_0, b_x, b_y, b_z, b_θ, b_g are the coefficients that vary for each unit. Units that were not tuned to any direction of movement velocity ($R^2 \leq 0.1$) were excluded from further processing. In matrix form, this relationship is written as:

$$F_{[t*n]} = V_{[t*d]} * B_{[d*n]} \quad (9)$$

where t is the number of decoding time points, n represents the number of neural features, and d represents the number of kinematic dimensions. The coefficient matrix B was solved using indirect optimum linear estimation (OLE) [64] and ridge regression [65].

$$B = (V^T V + \lambda_1 * I_{[d*d]}) \backslash V^T F \quad (10)$$

Where λ is the ridge regression optimization parameter and I is a d*d identity matrix. The following equation is satisfied by the decoding weights, W, that directly translate neuronal firing rates, F, to kinematic command signals, V.

$$V_{[t*d]} = F_{[t*n]} W_{[n*d]} \quad (11)$$

and were solved for using ridge regression and variance correction as below:

$$W = \left(B\, \Sigma_{[n*n]} B^T + \lambda_2 * I_{[d*d]}\right)^+ B\, \Sigma_{[n*n]} \quad (12)$$

where + represents the Moore-Penrose integral Pseudoinverse, and Σ is a diagonal matrix with values equal to the inverse variance of each neural unit's residuals. This effectively increases the contribution of units that carry more information regarding movement velocity and reduces the contribution of units that convey less or more variable information.

The weights of decoders were subsequently computed through indirect OLE. Afterward, the participant repeated the training task using the decoder trained from observation data, but the computer restricted the decoded movement velocities to those on the optimal path (The closest

possible routes between the start and target points). After completing this operation, a new decoder was trained using data from the second training batch. Every test day, a new decoder was trained, and all decoder training was completed without ICMS. The daily calibration process lasted around 15 minutes.

As depicted in Figure 21, each day began with the training of a new decoder utilizing a two-step procedure. First, using an observation paradigm, a primary decoder was trained. Next, the subject was assisted in controlling the robotic limb, and a new decoder was trained. After training the decoder, the participant completed a sequence task to provide control data from which we could determine the overall performance of the decoder. The individual then completed object transfer trials, followed by ARAT trials. On the first four trial days (717-729), ICMS was used to undertake these functional assessments. On the final four days of the trial (days 731-743), functional evaluations were performed without ICMS.

With ICMS, the median time spent attempting to grasp an object decreased by 66%, from 13.3 to 4.6 seconds. A one-by-one comparison of two identical actions reveals a decrease in the total time

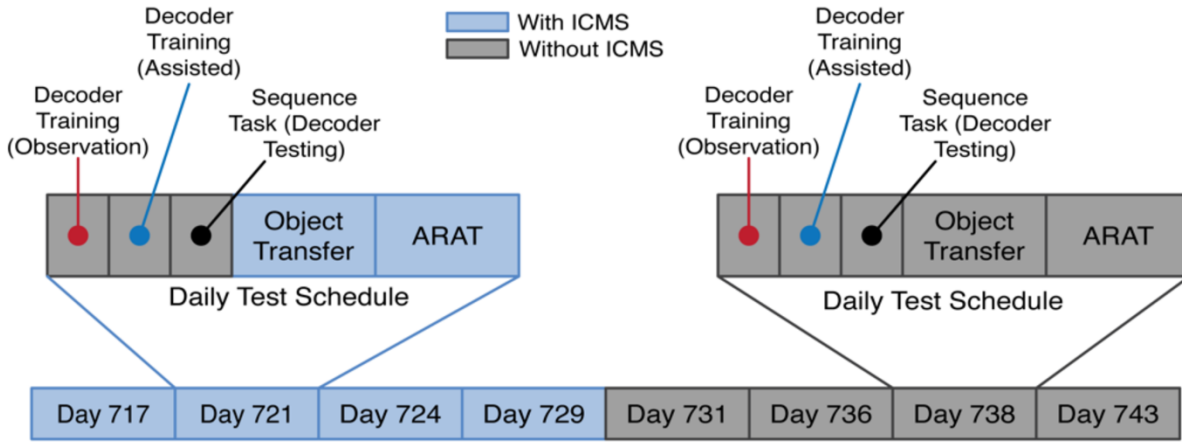

Figure 21 - Experimental timeline displaying the experimental schedule within and across days, noting when ICMS (blue) was utilized and when it was not (gray). Adapted from [57]

required to complete an action in 88% of cases. With ICMS, the reaching and transport portions required 25% less time on average. When a participant successfully grasps an object, they rarely lose it and all of the drops were caused by an unstable grasp, not because the participant's palm was opened.

To determine the baseline decoder performance accuracy in the absence of objects and, most crucially, in the absence of ICMS, a total of three sets of ten trials were conducted using the robotic limb. The performance of the decoder was evaluated on a daily basis using the MPL in a sequence task in which the goal was to acquire predetermined combinations of hand endpoint position, wrist orientation, and grasp posture. A trial was deemed successful if the participant was able to position the robotic hand within a 5 cm diameter target, orient the wrist to within ±0.25 radians, and adjust the grasp aperture to be at least 80% of the way to maximal flexion or extension of the digits being used.

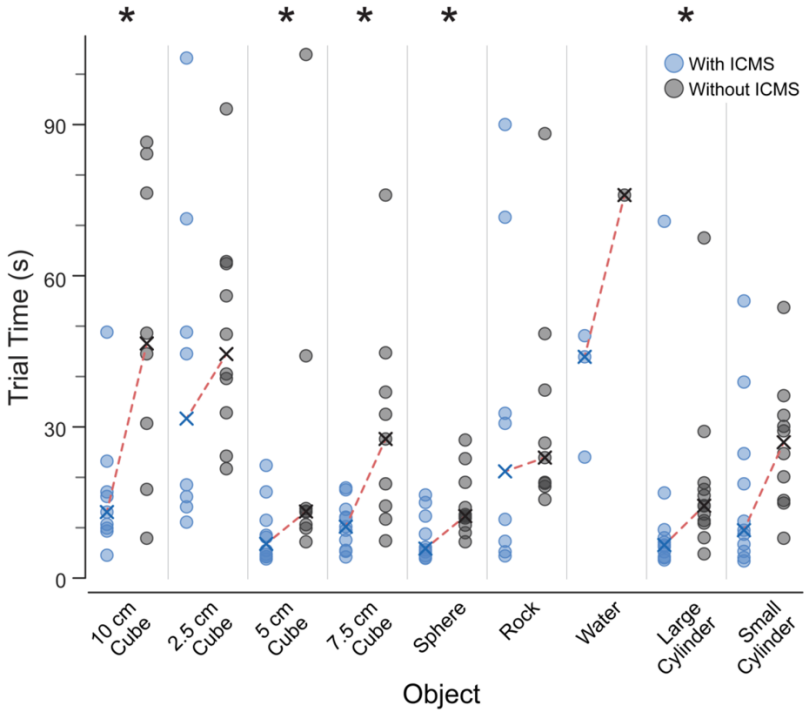

*Figure 22 - Effect of ICMS feedback on individual ARAT task item completion speeds. The * are showing the average of all the trials. Adapted from* [57]

Two tasks, a modified version of the ARAT (Figure 19a) and an item transfer task (Figure 19b), were performed with and without ICMS while vision feedback was always present.

3.8 Results

As previously noted, the system has been evaluated using the conventional ARAT test. Table 5 includes the median and interquartile range (IQR) timings for successful ARAT trials of each object, as well as the number of successful completions (N) with and without ICMS-induced tactile sensations. There were not enough successful water pouring efforts to determine the median and interquartile range, so the times for all successful attempts are reported. Based on the data presented on Table 5, it is clear that using ICMS significantly reduces the amount of time required to accomplish tasks compared to not using it. Figure 22 depicts the impact of ICMS feedback on ARAT task completion durations for specific objects to illustrate the variations between the presence and absence of the ICMS. Gray dots represent successful trial times without ICMS, while blue dots represent successful trial times with ICMS. The median trial times for each object or

Table 5 - The results of ARAT test: The median and interquartile range (IQR) timing and the number of successful completions (N) of ARAT test with and without ICMS

	Without ICMS			**With ICMS**		
Object	Median (s)	IQR (s)	N	Median (s)	IQR (s)	N
10 cm cube	46.6	24.2 – 80.3	8	13.1	9.8 – 18.6	9
2.5 cm cube	44.5	32.8 – 62.0	10	31.6	15.2 – 60.1	8
5 cm cube	13.2	10.3 – 29.0	8	6.8	4.3 – 11.5	10
7.5 cm cube	27.6	13.7 – 38.9	9	10.2	6.0 – 13.2	11
Sphere	12.3	10.9 – 17.8	11	5.9	4.4 – 12.3	10
Rock	24	18.7 – 40.1	9	21.2	6.3 – 52.2	8
Large Cylinder	14.4	11.2 – 18.3	12	6.6	4.5 – 9.2	11
Small Cylinder	27	15.4 – 32.3	10	9.5	5.7 – 23.2	11
Water Pouring		76	1	24.0, 43.9, 48.1		3

feedback paradigm are denoted with an “x”. For visualization purposes, median timings for each object are connected with a red line. When ICMS feedback was provided, five of the nine objects had significantly shorter trial times, while the remaining four had marginally faster trial times. Figure 23a depicts the ARAT scores for a previous study with occasional ICMS on the left, the current study with ICMS in the middle, and without ICMS on the right; when ICMS feedback was provided, the ARAT scores were significantly higher than in the current experiment without ICMS feedback with red lines representing the median score. Histogram of effective trial durations with (blue) and without (gray) ICMS tactile feedback is depicted on Figure 23b. With ICMS, median

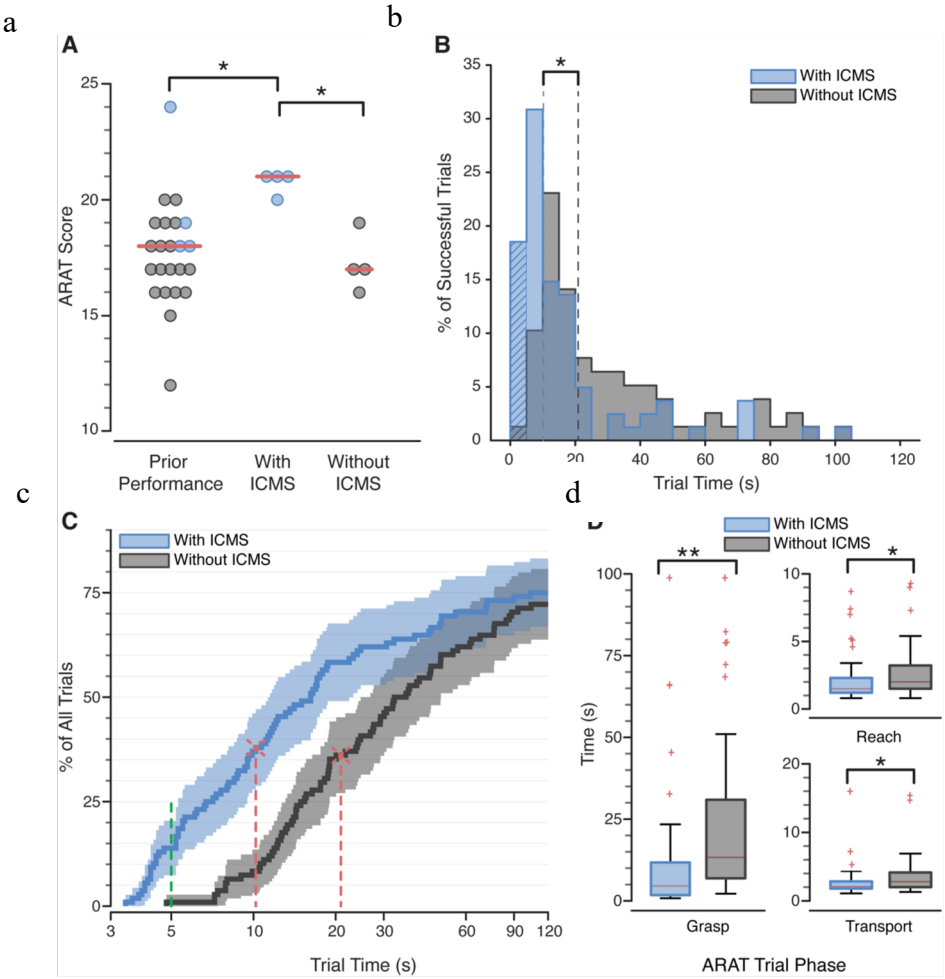

Figure 23 - The Results of ARAT test. Adapted from [57]

trial times (dotted lines) were substantially faster ($p<0.0001$). The bars with hatching represent trials completed in less than five seconds.

The empirical cumulative distribution function (ECDF) describes the distribution of a set of data points in terms of the proportion of data points that are less than or equal to a specified value. Figure 23c illustrates the empirical cumulative distribution of trial times, on a log-normalized axis, where the x-axis represents trial times on a logarithmic scale and the y-axis represents the proportion of trial times that are less than or equal to each value, with each step representing a change in the proportion of trial times. Figure 23d depicts the duration of each phase of the ARAT assignment. The red lines represent the medians, the box outlines represent the interquartile ranges, and the margins represent the range of the data excluding outliers (red '+'). All phases of a project were completed faster when ICMS feedback was provided (*$p<0.001$, **$p<0.0001$).

3.9 Conclusion

This study demonstrates that ICMS-induced tactile perceptions can enhance task performance to levels not previously observed and reduce reaching and grasping time in a manner comparable to that of natural tactile sensations during grasp. These performances appeared to be unrelated to practice. Artificial tactile sensations significantly improved performance, which will in the future substantially improve BCI performance. The results of this study are encouraging for helping individuals with SCI perform ADLs.

4 Real-time prediction of hand trajectory by ensembles of cortical neurons in primates

This study [66] describes an experiment in which MEAs were implanted in the brains of owl monkeys performing reaching and grasping actions in order to record their neural activity. This

neural data was used to train an algorithm capable of predicting the motion of the hand in real time. The system was trained using data from multiple cortical regions, and the researchers discovered how employing a collection of neurons from separate areas improved prediction accuracy. In addition, they discovered the algorithm's predictions were more accurate when a monkey was performing a simple activity as opposed to a complex one. Overall, the research demonstrates that ensembles of cortical neurons can be used to predict hand movements in real-time, which has implications for the development of brain prostheses for paralyzed individuals.

4.1 Implantation Procedure

Microelectrode arrays consisting of 16–32 Teflon-coated, stainless steel microwires (50 mm in diameter) were implanted into the cortical regions of two owl monkeys (Aotus trivirgatus). The premotor, primary motor, and posterior parietal cortical areas of owl monkeys were identified

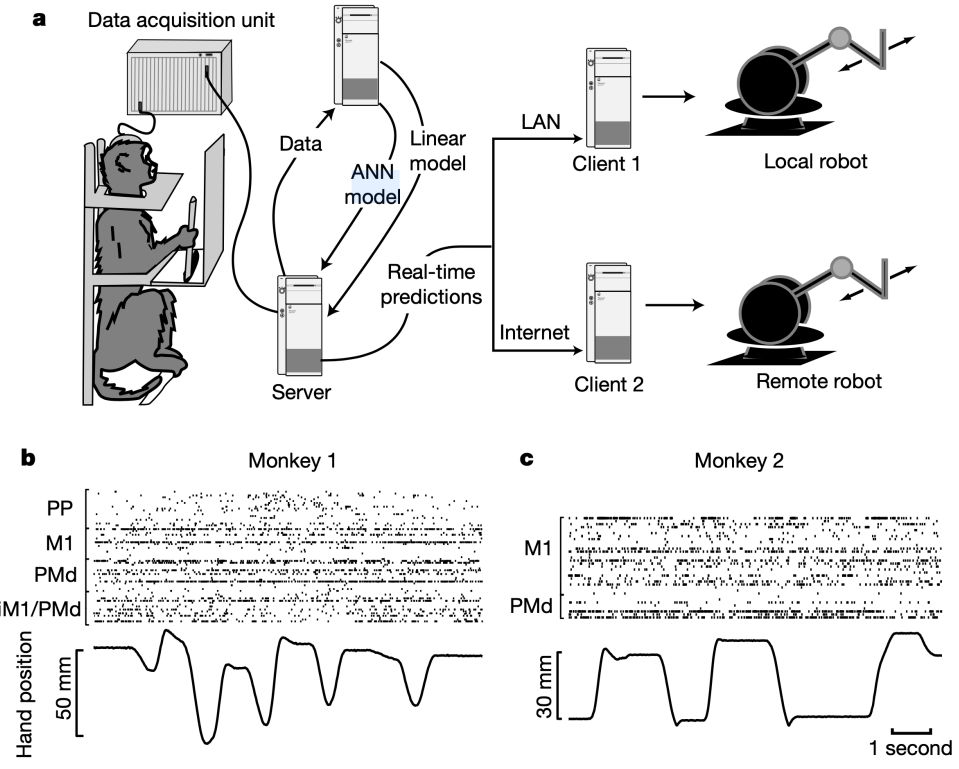

Figure 24 - A Schematic diagram of the experiment. Adapted from [66]

using stereotaxic coordinates, microstimulation maps, and intraoperative neural mapping recordings. During the implantation procedure, constant mechanical stimuli were applied to the arm, face, and legs, such as tapping the muscles and passively moving the joints, to locate the rostral and caudal borders of cortical regions from which somatosensory responses could be elicited while the monkey was under anesthesia. The implants were placed in the motor cortex prior to the rostral most limit of the somatosensory cortex and in the monkey's cortex immediately posterior to the caudal most limit of the somatosensory cortex once the boundaries of the motor cortex and the somatosensory cortex were determined. Through each implanted microwire, neural activities were recorded using a 96-channel neuron acquisition processor (MNAP, Plexon, Dallas, TX). Left dorsal premotor cortex (PMd, 16 wires) [67,68], left primary motor cortex (MI, 16 wires), left posterior parietal cortex (PP, 16 wires), right PMd and MI (32 wires), and right PP cortex (16 wires) were implanted with 96 microwaves in the first primate. Thirty-two microwires were implanted in the left PMd (16 wires) and left MI (16 wires) of the second primate. Figure 24b and c depict synchronized recordings from these regions as well as the hand position.

4.2 Tasks

Two owl monkeys were trained on two behavioral tasks. In the first task, the monkeys were instructed to center a manipulandum for a variable amount of time before moving it to a left or right target in response to a visual signal. They received a juice reward for correct responses. The position of the manipulandum was continuously recorded at 200 Hz using a precision potentiometer. In the second task, the monkeys were instructed to place their right hand on a platform attached to a chair at waist height. When an opaque barrier was removed, the monkeys were to pick a small piece of fruit randomly placed in one of four fixed target positions on a tray in front of them. The position and orientation of the wrist was continuously recorded in three-

dimensional space using a plastic strip containing numerous fiber optic sensors. The bending and twisting of the plastic strip altered light transmission through the fiber optic sensors (Shape Tape, Measurand, Inc., Fredericton, NB, Canada), providing a precise description of the wrist position in the x, y, and z dimensions. The resulting analog data was sampled at 200 Hz and converted to 3D arm trajectories. The arm trajectories were used to control the movements of local and remote robotic devices[69], as shown in Figure 24a.

4.3 Data Analysis

In this study, the "serial single-neuron recording" technique was used to characterize each individual neuron using recordings from a single or multiple microwires. This may entail employing signal processing techniques, such as spike sorting algorithms, to separate the electrical activity of individual neurons from the surrounding background noise or other neurons. In certain instances, researchers may perform serial single-neuron recordings by leaving the electrode in place for extended periods of time to record from the same neuron over the course of multiple sessions or even days. After identifying individual neurons, the firing rates of the neurons were extracted and inserted into the input matrix denoted as X(t).

Based on the input neural data, the limb position was predicted using first a linear model and then an ANN. The recording sessions with both primates were analyzed offline to determine how well a linear model can manage the predictions. To construct the linear model, which is an extension of simple linear regression, each column of the input matrix X(t) represents the discharges of individual neurons, and each row represents a time bin. Arm position denoted as Y(t) which is a single output vector in the case of a one-dimensional task and a three-column matrix in the case of a three-dimensional task. Significant coupling between inputs and outputs in this instance is frequently not restricted to synchronous observations but can occur over a range of time latency

or delay between signals. The expression for the linear association between neural discharges in X(t) and arm position in Y(t) is:

$$Y(t) = b + \sum_{u=-m}^{n} a(u)X(t-u) + \epsilon(t) \qquad (13)$$

In which:

- X(t) is neuronal firing in time t.
- Y(t) is arm trajectory in time t.
- a(u) is the weights required for fitting X(t) to Y(t) as a function of time lag u between inputs.and the outputs, it is called as impulse response function in this study, and
- b is the constant.

Equation (13) approximates the arm trajectory Y(t) by convolving X(t) with the a(u) which is referred to as impulse response function. In regression, the term b represents the Y-intercept. For one-dimensional motion, b is a single digit; for three-dimensional motion, b is a vector with three numbers, one for each dimension. $\epsilon(t)$, the final term in the equation, reflects the residual errors, or any fluctuation in Y(t) that cannot be explained by X(t).

The bounds of the time lag u should be selected so that the model includes time delays for which there is statistically significant coupling between the signals in X(t) and Y(t). The desired values of m and n may be calculated first using high numbers (e.g., 5-10 s) and subsequently refined by statistically analyzing impulse response functions across several data sets. In this study, it was found that the impulse response functions of some cortical neurons were statistically different from zero for delays of up to one second.

In this study, neuronal activities were regarded as discrete processes, but the location of the monkey's wrist in the three dimensions was deemed continuous. The hand trajectory data was

collected at a rate of 200 Hz, so 5 ms intervals of neural activity were chosen to correspond with this frequency. Calculating single neuron impulse response functions directly in the time domain at such a high temporal precision is computationally expensive. As a result, impulse response functions were calculated offline using the frequency-domain approach. First, the auto spectra for all input and output signals, as well as the cross spectra between all signal pairs, are computed using a Fast Fourier Transform (FFT) of synchronous segments of all signals and averaging over all segments in the data set used to build the model. This yields a spectral density matrix, $f(\lambda)$, which is the frequency-domain equivalent of the covariance matrix between all input and output variables.

The auto-spectrum is the spectral analysis of a signal's own power as a function of frequency. To calculate an auto spectrum using FFT, the FFT of the signal must first be calculated, followed by the magnitude squared of the FFT. The squared magnitude of the FFT produces the power spectrum of the signal, which reflects the frequency-dependent distribution of signal power. Cross-spectra is the spectral representation of the relationship between two signals as a function of frequency. A cross-spectrum is calculated by taking the FFT of both signals, multiplying them together, and then calculating the magnitude squared of the resulting product. The magnitude squared of the product yields the cross-spectrum of the two signals, which is the frequency-dependent distribution of the relationship between the two signals. The spectral density of a signal shows how its power or energy is spread across different frequencies.

The spectral density matrix $f(\lambda)$ is divisible into $f_{XX}(\lambda)$ and $f_{YX}(\lambda)$ where λ represents frequency, $f_{XX}(\lambda)$ describes the frequency-domain relationships among all X(t) inputs, $f_{YX}(\lambda)$ describes the relationships between outputs Y(t) and inputs X(t). Using inverse Fourier transforms, impulse response functions were derived from transfer functions. Using the sample means for each of the

input and output signals, the Y-intercept constants were finally determined. The following equation can be used to calculate the transfer functions between the frequency-domain analog of the input signals in X(t) and the output signals in Y(t):

$$A(\lambda) = f_{YX}(\lambda) f_{XX}(\lambda)^{-1} \quad (14)$$

where a matrix inverse is denoted by "to the power of minus unity." The gain and phase relationships between each pair of input-output signals are described as a function of frequency by the transfer functions in the matrix $A(\lambda)$. The impulse response function a(u) is also easily calculable using the inverse FFT algorithm because they are equivalent to the inverse Fourier transforms of the transfer functions. Finally, the relation estimates the Y-intercept constants b can be calculated as:

$$b = \overline{Y}(t) - A(0)\overline{X}(t) \quad (15)$$

where the overlines represent the sample means for every signal in X(t) and Y(t).

Coherence spectra were calculated to assess the coupling between individual neuron activity and arm position. As previously mentioned, spectra and cross-spectra for pairs of signals were calculated by Fourier transforming data segments and averaging over all available segments. A spectral resolution of 0.5 Hz was attained using 2-second segments. The cross-spectrum between the two is $f_{XY}(\lambda)$, the spectrum for a single neuron is $f_{XX}(\lambda)$, and the spectrum for the position is $f_{YY}(\lambda)$. The definition of the coherence spectrum is:

$$\left|R_{xy}(\lambda)\right|^2 = \frac{\left|f_{xy}(\lambda)\right|^2}{f_{xx}(\lambda) f_{yy}(\lambda)} \quad (16)$$

In other words, the coherence spectrum is the absolute squared cross-spectrum between the two signals, which is then normalized by the spectra of the two individual signals. On a scale that goes from zero to one, the coherence spectrum provides a description of the amount of linear coupling

that exists between the two signals. This description is based on the frequency. Utilizing standard practices, a statistical analysis of the significance of the coherence spectrum was carried out.

4.4 Real Time Data Analysis

There were two major obstacles to real time data processing. Always, there is a lag between arm position data Y(t) and neural activity X(t). Hence, for immediate prediction of Y(t), prior neural activities must be exploited to predict future arm motions. Given the limited processing resources available at the time (dual 800MHz Pentium III, PC-compatible microcomputer), it was essential to reduce the system's computational load. To simplify the process, instead of analyzing neural activity as a point process at 200 Hz, the number of threshold crossings in ten 100 ms bins was counted as shown in formula (17). To overcome the first challenge, the model was trained with the neural activity of the most recent ten time-bins, which will account for the most recent one second of data.

$$Y(t) = b + \sum_{u=0}^{9} a(u)X(t-u) + \epsilon(t) \qquad (17)$$

4.5 Artificial neural network model

In this study, the same data structure was utilized as in the real-time linear model, with inputs consisting of up to ten 100 ms divisions of neuronal discharge counts for each recorded neuron. The ANNs were feed-forward networks with a nonlinear hidden layer of units whose output function was tan-sigmoid, feeding a linear output layer that predicted the location signals [66,70]. The most successful approach for training the networks was discovered to be with one hidden layer comprising 15 to 20 units. They also used an early halting strategy to avoid data over-fitting. It involves stopping the training process early before the model has had a chance to overfit the training data. This can be done by monitoring the performance of the model on a validation set,

and stopping training when the performance on the validation set starts to decrease. Real-time predictions of hand position generated by ANNs were comparable to those generated by the linear approach, and the ANN model was often only marginally superior to the linear technique during offline analysis. The authors used an adaptive procedure to repeatedly fit the linear and ANN models during each recording session, with the models being computed using the most recent 10 minutes of recorded data. The outcomes of the models were sent over a typical Internet protocol server to computer clients operating robot arms in their lab and at the Massachusetts Institute of Technology (MIT). The robots were three-degree-of-freedom, high-precision manipulators, and their motions were recorded to determine the correctness of the arm trajectory signals provided by the models. The input command to the robots was the end-Cartesian effector's coordinates, and standard proportional derivative (PD) control of the end-effector coordinates was implemented in

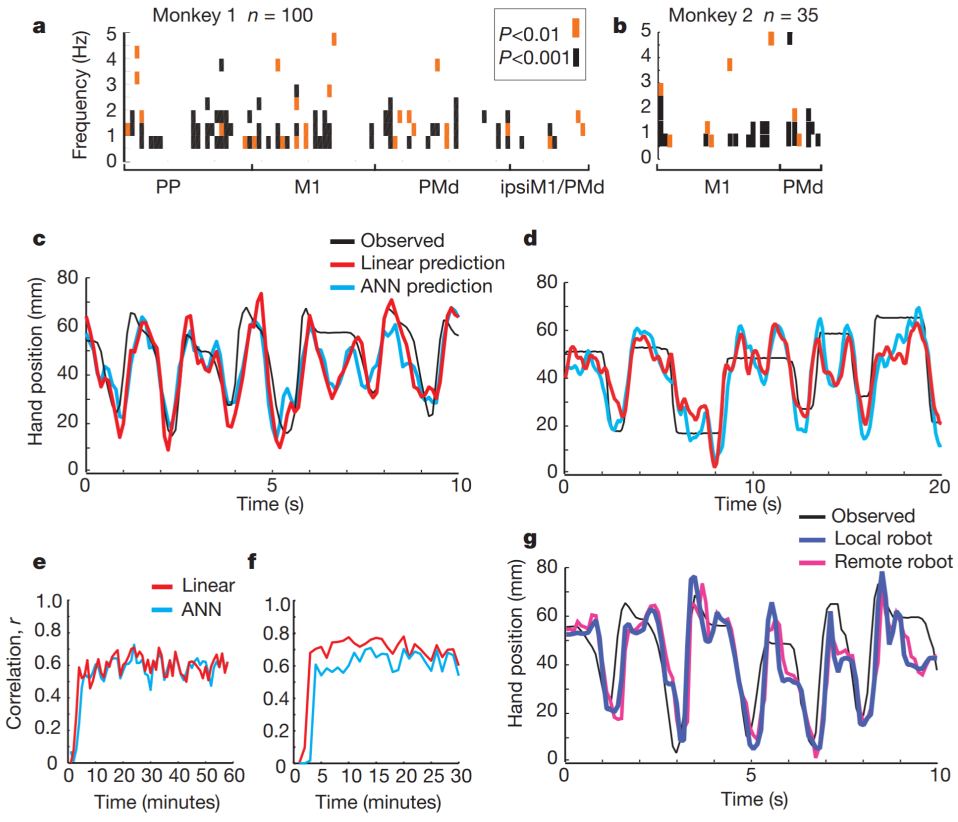

Figure 25 - Real-time control of 1D hand movements. Adapted from [66]

C++ with a new coordinate commanded every 100 ms based on the brain output of a monkey. As depicted in Figure 25e (monkey 1) and Figure 25f (monkey 2), the performance of both algorithms improved in the first few minutes of recordings for both primates, before reaching an asymptotic level that was maintained throughout the experiment. For several months, highly significant predictions of hand movement trajectories were derived for both primates.

4.6 Results

Coherence analysis revealed that the majority of single neurons from the recorded cortical regions exhibited significant correlations with both one-dimensional and three-dimensional hand trajectories, although the strength and frequency range of these correlations varied among cortical regions. Then, experiments were conducted to evaluate the real-time hand position prediction capabilities of linear and ANN algorithms. Despite the complexity of the trajectories, which involved varying starting positions and velocities, both algorithms produced highly accurate predictions for the one-dimensional movements of both primates. In one session, the activity of multiple neurons enabled monkey 1 to accomplish an average correlation coefficient of 0.61 between observed and predicted hand position, whereas a smaller sample of neurons produced an average correlation coefficient of 0.72 for monkey 2. The accuracy of predictions improved within the first few minutes of recordings and then stabilized at an asymptotic level that was maintained throughout the experiment, with significant predictions obtained in both primates over the course of several months. Figure 25c and d illustrate that there were no significant differences in performance between linear and ANN algorithms for either animal. (With linear prediction shown as the green line and ANN as the red line).

Throughout the recording sessions, both linear and ANN models were continuously updated to reduce the influence of dynamic variations in the coupling between neuronal activity and movement, as well as other non-stationary influences. This strategy significantly enhanced the ability to predict hand trajectories. For example, when predicting the last 10 minutes of 50-100 minute sessions, the adaptive algorithm outperformed a fixed model based on the first 10 minutes by 55% (median) in 20 sessions, and by 20% compared to a model based on the first 30-40 minutes of the session.

Later, the ability of the same cortical ensemble activity and models to predict the complex sequences of three-dimensional hand movements employed by primates in a food-reaching task (task 2) was investigated. These movements consisted of four phases: reaching for the meal, grasping it, bringing it to the mouth, and returning to the starting position. (Figure 26a and b). Due

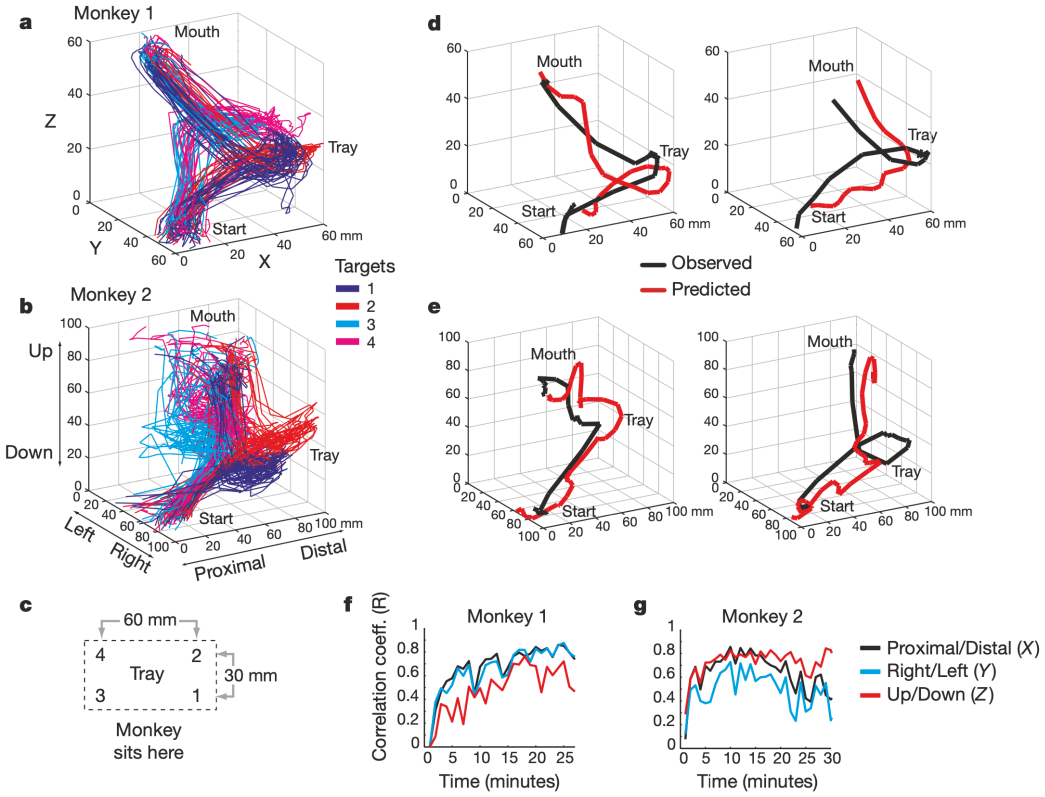

Figure 26 - Real-time control of 3D hand movements. Adapted from [66]

to the animals were not overtrained, their movement trajectories were highly variable. In the session depicted in Figure 26a (monkey 1), the hand trajectories displayed a dispersion of 7.0*7.5*6.0 cm (or 315 cm^3), whereas in Figure 26b (monkey 2), the dispersion was even greater, measuring 11.5*10.5*10.0 cm (or 1,207.5 cm^3).

The same linear and ANN models described above provided accurate predictions of three-dimensional hand trajectories in four distinct orientations during 25-60-minute experimental sessions (60-208 trials) for both animals. Figure 26d and e present examples of observed (black lines) and predicted (red lines) sequences of three-dimensional movements produced by monkey 1 and monkey 2, respectively. After initial advancements, asymptotic levels of prediction accuracy were achieved and maintained throughout the experiments for three-dimensional hand trajectories. (Figure 26f and g). The accuracy of three-dimensional predictions was found to be comparable to that of one-dimensional movements, with correlation coefficients (r) ranging from 0.54 to 0.77 for the x-, y-, and z-dimensions and averaged over 20-minute intervals, depending on the primate.

The results demonstrate the validity of the study's real-time methodology. The model parameters obtained from training with hand movements directed to one set of targets (e.g., targets on the right) reliably predicted hand trajectories directed to a different set of targets. (e.g., targets on the left). Similarly, accuracy was comparable when using parameters derived from movements to one set of targets to predict movements to the opposite set of targets (e.g., proximal to distal or vice versa). Using parameters trained with right movements to predict left movements, the correlations (r) between predicted and actual hand trajectories were 0.80, 0.70, and 0.67 for the x-, y-, and z-dimensions, respectively, in monkey 1. For macaque 2, the correlations for the x, y, and z dimensions were 0.68, 0.53, and 0.81, respectively. Similar accuracy was observed when predicting distal movements using proximal movement parameters and vice versa, with

correlations ranging from 0.69 to 0.81 for various dimensions and primates. These results indicate that the linear model used in the real-time approach is capable of generalizing across various hand movement directions, indicating its robustness and potential for practical applications in predicting hand trajectories in real-time tasks.

Neuron-dropping analysis is a method for determining the contribution of individual neurons to a specific neural computation or behavior. It entails systematically removing one neuron at a time from the neural population being analyzed, followed by measuring the effect of the neuron's absence on the performance or behavior of the entire system. Typically, neuron-dropping analysis is conducted by simulating the neural system or model with and without every neuron and comparing the performance or behavior in both cases. This enables the determination of the relative significance of each neuron in the computation or behavior of interest.

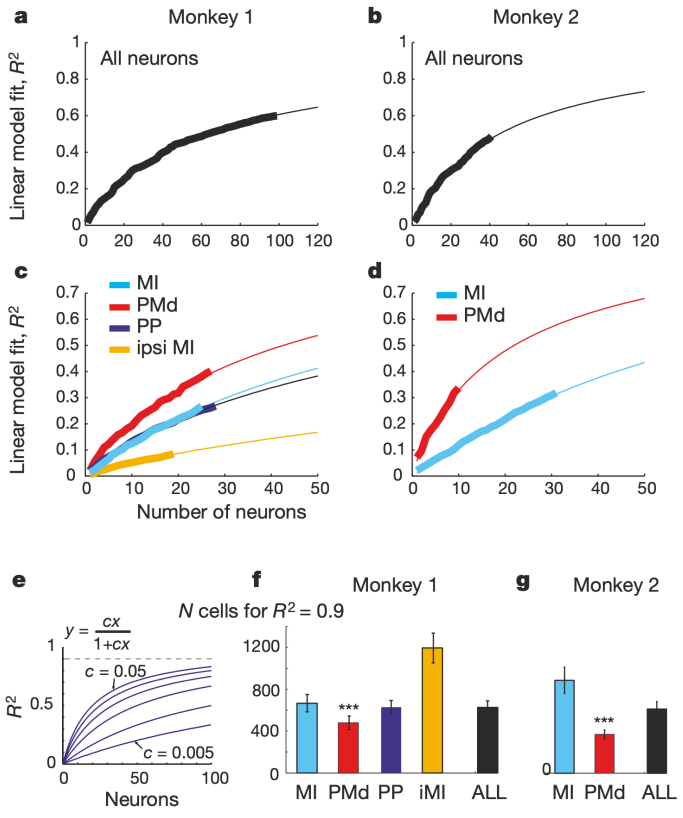

Figure 27 - Neuro Dropping Analysis. Adapted from [66]

Using a neuron-dropping analysis, the study examined the contributions of various cortical regions to the prediction of one-dimensional hand movements. This analysis involved determining the average effect of removing individual neurons from the neuronal population utilized in each real-time session. The analysis was conducted offline separately for each cortical area, as well as for all cortical areas combined, neuron-dropping curves were obtained, and simple hyperbolic functions were used to fit them with. The results indicated that hyperbolic functions could accurately fit the curves derived from the neuron-dropping analysis, with correlation coefficients (r) ranging from 0.9966 to 0.9996, using both the linear and ANN models for both animals. Figure 27a-e illustrates typical neuron-dropping curves and their hyperbolic fits. Extrapolations of the hyperbolic curves revealed that theoretically, 90% accurate real-time prediction of one-dimensional hand movements could be attained by applying the linear model to a specified number of neurons in each cortical region. As shown in Figure 27f and g for monkey one 480±65.7 PMd, 666±83.0 M1, 629±64.2 PP and 1,195±142 ipsilateral MI/PMd neurons is required to get 90% of accuracy. For monkey 2, the same level of accuracy would require either 376±42.3 PMd neurons or 869±127.4 MI neurons. Theoretically, significantly fewer PMd (red) neurons would be required in both primates to attain the same level (90%) of one-dimensional hand movement prediction accuracy, i.e., PMd neurons contributed the most to the predictions on average. MI (light blue) and PP (dark blue) ensembles contributed comparably less variance, whereas ipsilateral MI cortical neurons accounted for the least variance. (Yellow line). When all recorded cortical neurons were combined, extrapolation of hyperbolic functions yielded identical 90% prediction accuracy estimates for both primates (monkey 1, 625±64 neurons; monkey 2, 619±73 neurons).

This study proves that as it was hypothesized the motor control signals for arm movements occur simultaneously in large regions of the frontal and parietal cortices, and that each of these cortical

regions may produce hand trajectory signals in real time. Nonetheless, the estimated neuronal sample necessary to predict hand trajectories using a single cortical region may vary, indicating functional specializations of these regions. According to previous findings, the activity in these cortical regions is influenced by motor parameters and other factors, such as visual information or motor periphery.

This study demonstrated that neural ensemble activity recorded from numerous cortical regions can be used to generate both one-dimensional and three-dimensional signals for controlling robot movement in real time. This real-time approach, unlike previous offline algorithms, did not rely on a priori presumption regarding the physiological properties of solitary neurons or the homogeneity of the neuronal population sample. Instead, random samples of cortical neurons were used to predict arm movements accurately. These findings support the notion that motor signals from ensembles of cortical neurons could potentially be used to govern the movements of prosthetic limbs over the long term. A combination of denser multi-wire arrays and implantable integrated circuits for real-time signal processing could form the foundation of a brain-machine interface for paralyzed patients to control prosthetic limbs voluntarily.

5 Conclusion

Each of the three papers we examined had been selected for a particular purpose. The first one was chosen as one of the most recent and innovative BCI systems with the highest reported degrees of freedom (31 character). The second one was chosen due to its uniqueness as a close-loop BCI system that can provide tactile feedback to the individual, the second system. The third paper was chosen to compare the methodologies of a pioneering paper from the past with those of contemporary research.

The methods of older studies are different from those of modern studies in two major ways. The first difference is that older papers looked at different parts of the brain to figure out what each part did. In newer studies, however, since most parts of the brain have already been studied in depth, researchers are less concerned with figuring out what each part does and tend to carefully implant MEAs in parts of the brain that have already been studied. The second distinction is that older studies use spike sorting to ascertain the activities of all the neurons adjacent to the electrodes, whereas newer approaches tend to use threshold crossing rates of neural activities without regard for individual neurons. This is possible due to the development of MEAs with reduced cross-section areas and shortened distances between electrodes, as well as the improvement of computer and machine learning algorithm processing capabilities. In the meantime, because thresholding is computationally cheaper than spike sorting, it enables the use of intricate machine learning algorithms in real time.

Since data was always recorded at sampling rates such as 30 kSps on each electrode, data transmission to the outside of the brain was always a challenge. Using any wireless system would require a large amount of power, producing excess heat that could damage the tissues. With basic circuitry, it is possible to threshold neural activity and only transmit threshold crossing rates out of the brain using thresholding techniques. The new BCI system implanted by Neuralink [71] in primates appears to be based on this concept, which transmits data over Bluetooth at low data rates. Considering the advancements in hardware and machine learning algorithms, it is beneficial to conduct research on stroke patients. As mentioned in chapter 1, the majority of people with UE disability are stroke patients, and there is a growing demand for BCI devices that are compatible with stroke patients.

6 References

1. Khantan M, Avery M, Aung P, Zarin R, Hammelef E, Shawki N, et al. The NuroSleeve, A User-Centered 3D Printed Orthosis and Functional Electrical Stimulation System for Individuals with Upper Extremity Impairment. 2023.

2. O'Neill C, Proietti T, Nuckols K, Clarke ME, Hohimer CJ, Cloutier A, et al. Inflatable Soft Wearable Robot for Reducing Therapist Fatigue during Upper Extremity Rehabilitation in Severe Stroke. IEEE Robot Autom Lett. Institute of Electrical and Electronics Engineers Inc.; 2020;5:3899–906.

3. Allison R, Shenton L, Bamforth K, Kilbride C, Richards D. Incidence, Time Course and Predictors of Impairments Relating to Caring for the Profoundly Affected arm After Stroke: A Systematic Review. Physiotherapy Research International. John Wiley and Sons Ltd; 2016;21:210–27.

4. Dunkelberger N, Schearer EM, O'Malley MK. A review of methods for achieving upper limb movement following spinal cord injury through hybrid muscle stimulation and robotic assistance. Exp Neurol. Academic Press Inc.; 2020.

5. What is Muscular Dystrophy? | CDC [Internet]. [cited 2022 Oct 14]. Available from: https://www.cdc.gov/ncbddd/musculardystrophy/facts.html

6. Garcia-Garcia LA, Rodríguez-Salvador M. Competitive and technology intelligence to reveal the most influential authors and inter-institutional collaborations on additive manufacturing for hand orthoses. 2018.

7. Virani SS, Alonso A, Aparicio HJ, Benjamin EJ, Bittencourt MS, Callaway CW, et al. Heart Disease and Stroke Statistics - 2021 Update: A Report From the American Heart Association. Circulation. Lippincott Williams and Wilkins; 2021. p. E254–743.

8. Norouzi-Gheidari N, Hernandez A, Archambault PS, Higgins J, Poissant L, Kairy D. Feasibility, safety and efficacy of a virtual reality exergame system to supplement upper extremity rehabilitation post-stroke: A pilot randomized clinical trial and proof of principle. Int J Environ Res Public Health. MDPI AG; 2020;17.

9. de Miguel-Rubio A, Dolores Rubio M, Alba-Rueda A, Salazar A, Moral-Munoz JA, Lucena-Anton D. Virtual reality systems for upper limb motor function recovery in patients with spinal cord injury: Systematic review and meta-analysis. JMIR Mhealth Uhealth. JMIR Publications Inc.; 2020.

10. Yozbatiran N, Francisco GE. Robot-assisted Therapy for the Upper Limb after Cervical Spinal Cord Injury. Phys Med Rehabil Clin N Am. W.B. Saunders; 2019. p. 367–84.

11. Block VAJ, Pitsch E, Tahir P, Cree BAC, Allen DD, Gelfand JM. Remote physical activity monitoring in neurological disease: A systematic review. PLoS One. Public Library of Science; 2016.

12. Singh H, Unger J, Zariffa J, Pakosh M, Jaglal S, Craven BC, et al. Robot-assisted upper extremity rehabilitation for cervical spinal cord injuries: a systematic scoping review. Disabil Rehabil Assist Technol. Taylor and Francis Ltd; 2018. p. 704–15.

13. Kumar S, Sahin F. A framework for a real time intelligent and interactive Brain Computer Interface. Computers & Electrical Engineering [Internet]. 2015;43:193–214. Available from: https://www.sciencedirect.com/science/article/pii/S0045790615001135

14. Shih JJ, Krusienski DJ, Wolpaw JR. Brain-Computer Interfaces in Medicine. Mayo Clin Proc. 2012;87:268–79.

15. Kayagil TA, Bai O, Henriquez CS, Lin P, Furlani SJ, Vorbach S, et al. A binary method for simple and accurate two-dimensional cursor control from EEG with minimal subject training. J Neuroeng Rehabil. 2009;6.

16. McFarland DJ, Krusienski DJ, Sarnacki WA, Wolpaw JR. Emulation of computer mouse control with a noninvasive brain-computer interface. J Neural Eng. 2008;5:101–10.

17. Wolpaw JR, Mcfarland DJ. Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans [Internet]. 2004. Available from: www.pnas.orgcgidoi10.1073pnas.0403504101

18. Müller-Putz GR, Scherer R, Pfurtscheller G, Rupp R. EEG-based neuroprosthesis control: A step towards clinical practice. Neurosci Lett. 2005;382:169–74.

19. Neuper C, Müller-Putz GR, Scherer R, Pfurtscheller G. Chapter 25 Motor imagery and EEG-based control of spelling devices and neuroprostheses. Prog Brain Res. 2006. p. 393–409.

20. Cincotti F, Mattia D, Aloise F, Bufalari S, Schalk G, Oriolo G, et al. Non-invasive brain-computer interface system: Towards its application as assistive technology. Brain Res Bull. 2008;75:796–803.

21. Pfurtscheller G, Guger C, Mu G, Ller È, Krausz G, Neuper C. Brain oscillations control hand orthosis in a tetraplegic [Internet]. Available from: www.elsevier.com/locate/neulet

22. Pfurtscheller G, Müller GR, Pfurtscheller J, Gerner HJ, Rupp R. “Thought” - Control of functional electrical stimulation to restore hand grasp in a patient with tetraplegia. Neurosci Lett. Elsevier Ireland Ltd; 2003;351:33–6.

23. Meng J, Zhang S, Bekyo A, Olsoe J, Baxter B, He B. Noninvasive Electroencephalogram Based Control of a Robotic Arm for Reach and Grasp Tasks. Sci Rep. Nature Publishing Group; 2016;6.

24. Tariq M, Trivailo PM, Simic M. EEG-Based BCI Control Schemes for Lower-Limb Assistive-Robots. Front Hum Neurosci. Frontiers Media S.A.; 2018.

25. Galán F, Nuttin M, Lew E, Ferrez PW, Vanacker G, Philips J, et al. A brain-actuated wheelchair: Asynchronous and non-invasive Brain-computer interfaces for continuous control of robots. Clinical Neurophysiology. 2008;119:2159–69.

26. Tanaka K, Matsunaga K, Wang HO. Electroencephalogram-based control of an electric wheelchair. IEEE Transactions on Robotics. 2005;21:762–6.

27. Mestais CS, Charvet G, Sauter-Starace F, Foerster M, Ratel D, Benabid AL. WIMAGINE: Wireless 64-Channel ECoG Recording Implant for Long Term Clinical Applications. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2015;23:10–21.

28. Wang W, Collinger JL, Degenhart AD, Tyler-Kabara EC, Schwartz AB, Moran DW, et al. An Electrocorticographic Brain Interface in an Individual with Tetraplegia. PLoS One. 2013;8.

29. Williams JJ, Rouse AG, Thongpang S, Williams JC, Moran DW. Differentiating closed-loop cortical intention from rest: building an asynchronous electrocorticographic BCI. J Neural Eng [Internet]. IOP Publishing; 2013;10:46001. Available from: https://dx.doi.org/10.1088/1741-2560/10/4/046001

30. Moran D. Evolution of brain-computer interface: Action potentials, local field potentials and electrocorticograms. Curr Opin Neurobiol. 2010. p. 741–5.

31. Willett FR, Avansino DT, Hochberg LR, Henderson JM, Shenoy K v. High-performance brain-to-text communication via handwriting. Nature. Nature Research; 2021;593:249–54.

32. NeuroPort Electrode Instructions for Use. 2022 [cited 2022 Nov 11]; Available from: www.blackrockneurotech.com

33. Blackrock Neurotech - Blackrock Neurotech [Internet]. [cited 2022 Nov 11]. Available from: https://blackrockneurotech.com/

34. OptiTrack - V120:Trio - An optical tracking system in a single, plug-and-play package [Internet]. [cited 2022 Nov 12]. Available from: https://optitrack.com/cameras/v120-trio/

35. Trautmann EM, Stavisky SD, Lahiri S, Ames KC, Kaufman MT, O'Shea DJ, et al. Accurate Estimation of Neural Population Dynamics without Spike Sorting. Neuron. Cell Press; 2019;103:292-308.e4.

36. Todorova S, Sadtler P, Batista A, Chase S, Ventura V. To sort or not to sort: the impact of spike-sorting on neural decoding performance. J Neural Eng. 2014;11:056005.

37. Christie BP, Tat DM, Irwin ZT, Gilja V, Nuyujukian P, Foster JD, et al. Comparison of spike sorting and thresholding of voltage waveforms for intracortical brain–machine interface performance. J Neural Eng. 2015;12:016009.

38. Chestek CA, Gilja V, Nuyujukian P, Foster JD, Fan JM, Kaufman MT, et al. Long-term stability of neural prosthetic control signals from silicon cortical arrays in rhesus macaque motor cortex. J Neural Eng. 2011;8:045005.

39. Poole B, Williams AH, Maheswaranathan N, Yu BM, Santhanam G, Ryu SI, et al. Time-warped PCA : simultaneous alignment and dimensionality reduction of neural data. 2016.

40. Williams AH, Poole B, Maheswaranathan N, Dhawale AK, Fisher T, Wilson CD, et al. Discovering Precise Temporal Patterns in Large-Scale Neural Recordings through Robust and Interpretable Time Warping. Neuron. 2020;105:246-259.e8.

41. GitHub - ganguli-lab/twpca: Time-warped principal components analysis (twPCA) [Internet]. [cited 2022 Nov 11]. Available from: https://github.com/ganguli-lab/twpca

42. Young S, Evermann G, Hain T, Kershaw D, Moore G, Odell J, et al. The HTK Book. 1995.

43. Web 1T 5-gram Version 1 - Linguistic Data Consortium [Internet]. [cited 2022 Nov 11]. Available from: https://catalog.ldc.upenn.edu/LDC2006T13

44. Degenhart AD, Bishop WE, Oby ER, Tyler-Kabara EC, Chase SM, Batista AP, et al. Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. Nat Biomed Eng. 2020;4:672–85.

45. Downey JE, Schwed N, Chase SM, Schwartz AB, Collinger JL. Intracortical recording stability in human brain–computer interface users. J Neural Eng. 2018;15:046016.

46. Jarosiewicz B, Sarma AA, Bacher D, Masse NY, Simeral JD, Sorice B, et al. Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. Sci Transl Med. 2015;7.

47. Downey JE, Schwed N, Chase SM, Schwartz AB, Collinger JL. Intracortical recording stability in human brain–computer interface users. J Neural Eng. 2018;15:046016.

48. Jarosiewicz B, Sarma AA, Bacher D, Masse NY, Simeral JD, Sorice B, et al. Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. Sci Transl Med. 2015;7.

49. Jarosiewicz B, Sarma AA, Bacher D, Masse NY, Simeral JD, Sorice B, et al. Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. Sci Transl Med. 2015;7.

50. Sussillo D, Stavisky SD, Kao JC, Ryu SI, Shenoy K v. Making brain–machine interfaces robust to future neural variability. Nat Commun. 2016;7:13749.

51. Degenhart AD, Bishop WE, Oby ER, Tyler-Kabara EC, Chase SM, Batista AP, et al. Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. Nat Biomed Eng. 2020;4:672–85.

52. Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I. Language Models are Unsupervised Multitask Learners. 2019.

53. GitHub - openai/gpt-2: Code for the paper “Language Models are Unsupervised Multitask Learners” [Internet]. [cited 2022 Nov 16]. Available from: https://github.com/openai/gpt-2

54. Collinger JL, Wodlinger B, Downey JE, Wang W, Tyler-Kabara EC, Weber DJ, et al. High-performance neuroprosthetic control by an individual with tetraplegia. The Lancet. 2013;381:557–64.

55. Gilja V, Nuyujukian P, Chestek CA, Cunningham JP, Yu BM, Fan JM, et al. A high-performance neural prosthesis enabled by control algorithm design. Nat Neurosci. 2012;15:1752–7.

56. Gilja V, Pandarinath C, Blabe CH, Nuyujukian P, Simeral JD, Sarma AA, et al. Clinical translation of a high-performance neural prosthesis. Nat Med. 2015;21:1142–5.

57. Flesher SN, Downey JE, Weiss JM, Hughes CL, Herrera AJ, Tyler-Kabara EC, et al. A brain-computer interface that evokes tactile sensations improves robotic arm control. Science (1979). 2021;372:831–6.

58. Perry BN, Moran CW, Armiger RS, Pasquina PF, Vandersea JW, Tsao JW. Initial Clinical Evaluation of the Modular Prosthetic Limb. Front Neurol. 2018;9.

59. Yozbatiran N, Der-Yeghiaian L, Cramer SC. A standardized approach to performing the action research arm test. Neurorehabil Neural Repair. 2008;22:78–90.

60. Lyle RC. A performance test for assessment of upper limb function in physical rehabilitation treatment and research. International Journal of Rehabilitation Research. 1981;4:483–92.

61. Collinger JL, Wodlinger B, Downey JE, Wang W, Tyler-Kabara EC, Weber DJ, et al. High-performance neuroprosthetic control by an individual with tetraplegia. The Lancet. 2013;381:557–64.

62. Flesher SN, Collinger JL, Foldes ST, Weiss JM, Downey JE, Tyler-Kabara EC, et al. Intracortical microstimulation of human somatosensory cortex. Sci Transl Med. 2016;8.

63. Penfield W, Boldrey E. Somatic motor and sensory representation in the cerebral cortex of man as studied by electrical stimulation. Brain [Internet]. 1937;60:389–443. Available from: https://doi.org/10.1093/brain/60.4.389

64. Wang W, Chan SS, Heldman DA, Moran DW. Motor Cortical Representation of Position and Velocity During Reaching. J Neurophysiol. 2007;97:4258–70.

65. Marquardt DW. Generalized Inverses, Ridge Regression, Biased Linear Estimation, and Nonlinear Estimation. Technometrics. 1970;12:591.

66. Wessberg J, Stambaugh CR, Kralik JD, Beck PD, Laubach M, Chapin JK, et al. Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. Nature. 2000;408:361–5.

67. Preuss TM, Stepniewska I, Kaas JH. Movement representation in the dorsal and ventral premotor areas of owl monkeys: A microstimulation study. J Comp Neurol. 1996;371:649–76.

68. Stepniewska I, Preuss TM, Kaas JH. Architectionis, somatotopic organization, and ipsilateral cortical connections of the primary motor area (M1) of owl monkeys. J Comp Neurol. 1993;330:238–71.

69. Salisbury JK, Srinivasan MA. Phantom-based haptic interaction with virtual objects. IEEE Comput Graph Appl. 1997;17:6–10.

70. Nicolelis MAL, Ghazanfar AA, Stambaugh CR, Oliveira LMO, Laubach M, Chapin JK, et al. Simultaneous encoding of tactile information by three primate cortical areas. Nat Neurosci. 1998;1:621–30.

71. Pager Plays MindPong - Neuralink [Internet]. [cited 2023 Apr 9]. Available from: https://neuralink.com/blog/pager-plays-mindpong/