Published in final edited form as:

J Neural Eng. 2017 August ; 14(4): 046025. doi:10.1088/1741-2552/aa7525.

Optimizing the stimulus presentation paradigm design for the P300-based brain-computer interface using performance prediction

B. O. Mainsah¹, G. Reeves^{1,2}, L. M. Collins^{1,3,*}, and C. S. Throckmorton¹

¹Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA

²Department of Statistical Science, Duke University, Durham, NC, USA

³Department of Biomedical Engineering, Duke University, Durham, NC, USA

Abstract

Objective—The role of a brain-computer interface (BCI) is to discern a user's intended message by extracting and decoding relevant information from brain signals. Stimulus-driven BCIs, such as the P300 speller, rely on detecting event-related potentials (ERPs) in response to a user attending to relevant or target stimulus events. However, this process is error-prone because the ERPs are embedded in noisy electroencephalography (EEG) data, representing a fundamental problem in communication of the uncertainty in the information that is received during noisy transmission. A BCI can be modeled as a noisy communication system and an information-theoretic approach can be exploited to design a stimulus presentation paradigm to maximize the information content that is presented to the user. However, previous methods that focused on designing error-correcting codes failed to provide significant performance improvements due to underestimating the effects of psycho-physiological factors on the P300 ERP elicitation process and a limited ability to predict online performance with their proposed methods. Maximizing the information rate favors the selection of stimulus presentation patterns with increased target presentation frequency, which exacerbates refractory effects and negatively impacts performance within the context of an oddball paradigm. An information-theoretic approach that seeks to understand the fundamental trade-off between information rate and reliability is desirable.

Approach—We developed a performance-based paradigm (PBP) by tuning specific parameters of the stimulus presentation paradigm to maximize performance while minimizing refractory effects. We used a probabilistic-based performance prediction method as an evaluation criterion to select a final configuration of the PBP.

Main results—With our PBP, we demonstrate statistically significant improvements in online performance, both in accuracy and spelling rate, compared to the conventional row-column paradigm.

Significance—By accounting for refractory effects, an information-theoretic approach can be exploited to significantly improve BCI performance across a wide range of performance levels.

*Corresponding author: leslie.collins@duke.edu.

Keywords

Brain-computer interface (BCI); P300 speller; Stimulus presentation paradigm; Information theory; Combinatorial problem; Codebook design; Performance prediction; Dynamic stopping

1 Introduction

Brain-computer interfaces (BCIs) can restore control or communication abilities to individuals with severe neuromuscular limitations due to neuronal diseases, stroke or traumatic brain injury [1, 2]. BCIs operate by monitoring electro-physiological signals, invasively or non-invasively, from different areas of the brain, thereby bypassing the brain's traditional pathways of nerves to effector muscle organs. These electro-physiological signals are processed in real time to extract features that help discern the user's intent and this information is translated into executable commands to control different types of devices.

One of the most commonly researched BCIs for communication augmentation are P300-based BCIs. The P300 speller, initially developed by Farwell and Donchin [3], enables users to make selections from a set of choices such as characters or icons that convey a desired intent or action. The P300 speller relies predominantly on detecting event-related potentials (ERPs) that are embedded in electroencephalography (EEG) data. These ERPs are elicited as a function of a user's uncertainty regarding stimulus events in either an acoustic, a tactile or a visual oddball recognition task [4]: the random occurrence of rare stimulus events, denoted as *oddball* or *target* stimuli, within a series of more frequent, or *non-target* stimulus events. Ideally, the presentation of the target stimulus event elicits a distinct ERP response that includes a large positive deflection in the EEG signal termed the P300 signal.

In a typical visual P300 speller, a user selectively attends to a desired or target character while groups of characters are sequentially illuminated on a screen [3]. The EEG responses to the stimulus events are then analyzed by an automated algorithm to discern the user's target character by using a classifier to distinguish between non-target and target stimulus events. Different classification techniques have been investigated to improve BCI performance [5, 6, 7], with linear discriminant analysis methods usually being preferred due to lower computational complexities and reduced classifier training times that are more suited for online BCI implementation. Due to the low signal-to-noise ratio (SNR) of the elicited ERPs that are embedded in noisy EEG signals, data are collected from multiple presentations of a potential target character to increase the SNR of the P300 ERP and improve selection accuracy [1]. The conventional method of data collection to increase ERP SNRs has been to collect a fixed amount of data, termed static stopping. Recently, there has been a shift towards dynamic stopping algorithms, where the amount of data collected is varied based on acute changes in signal quality [8, 9, 10]. Some BCI algorithms in the literature also exploit the predictability of language to inform the BCI selection process by using statistical language models to identify likely characters based on a user's spelling history [8, 11].

The stimulus presentation paradigm defines the presentation order of all of the character choices. While some P300 spellers use single character presentation, e.g. rapid serial visual

presentation speller [12], the majority of spellers present a group of characters in a single stimulus event in order to increase the character presentation rate and spelling speed. For example, with a user interface that has a grid layout, a simple method to group characters is by the rows and columns of the grid. This is the basis for the row-column paradigm (RCP) [3], which is the most commonly used mode of stimulus presentation in the literature [13]. However, presenting groups of characters increases the likelihood of selection errors because of the possible correlation in the cumulative EEG responses associated with the character presentations, especially for characters that are presented together often [14, 15]. In addition, there are non-linearities in the EEG responses to target stimulus events due to *refractory effects*, where the P300 ERP SNR is reduced due to the short time interval between target character presentations [16, 17]. For the RCP, it is possible for a target character to be presented successively due to the randomized order of the flash group presentations. It is also possible that with a short time interval between target character presentations, a user might miss a successive target stimulus event, especially if the stimulus presentation frequency is high.

Re-designing the stimulus presentation paradigm has the potential to minimize selections errors that result from refractory effects or grouping characters for presentation. Some approaches focus on enhancing the ERP responses to the target stimulus events by optimizing stimulation parameters such as the stimulus presentation rate [18], stimulus duration and inter-stimulus interval [19, 20], interface size [20], and stimulation intensity [21]. Other approaches focus on modifying cosmetic aspects of the visual interface to either increase user focus or minimize distractions, e.g. region-based [22] or eye-gaze independent interfaces [23]. Some methods incorporate salient elements during stimulus presentation, e.g. the use of color, shapes or faces, to elicit other ERPs that can enhance performance [24, 25, 26]. To mitigate refractory effects, some methods impose a minimum time interval between a character's presentation [15, 27, 28, 29, 30].

With all of these previous approaches, including the RCP, the generation and presentation of flash groups are randomized with limited consideration of methods for maximizing the information content that is presented to the user in order to improve performance. As an alternative to random character presentations, a BCI can be modeled as a communication system with a noisy channel, which provides a principled framework for the design of the flash groups in terms of information that is presented to the user. Coding theory [31, 32] provides a method for encoding information for efficient communication in spite of noisy channel transmission. Some studies have exploited coding theory to design stimulus presentation patterns for the P300 speller [33, 34, 35, 36]. However, in online testing, the proposed stimulus paradigms in [33, 34, 35, 36] resulted in similar or worse performance than the RCP, possibly due to underestimating the negative impact of refractory effects or the limited ability to pre-assess performance with a given stimulus paradigm configuration prior to online testing.

We hypothesize that the benefits of exploiting an information-theoretic approach can be obtained by taking into account the transmission dynamics of the noisy communication channel. Within a noisy channel framework, we present a new method to design a stimulus presentation paradigm that maximizes selection performance while minimizing refractory

effects, by tuning specific stimulus presentation parameters to positively impact performance. In addition, we optimize the design process for a dynamic stopping (DS) algorithm [37], as a DS criterion provides the flexibility to vary the amount of data collection based on the current ERP SNR conditions to improve performance [8, 9, 10]. Also, rather than inferring potential performance improvements with a given stimulus presentation configuration during the design process, we utilize a performance prediction method [38] to compare different configurations. In this online study, we compare a configuration of the stimulus presentation paradigm developed using our performance-based approach to the conventional RCP.

2 Theoretical Model

The P300 speller relies on detecting transient ERPs, embedded in noisy EEG data, in response to a user attending to target stimulus events. The ERP elicitation process is affected by several psycho-physiological factors, including the properties of the eliciting stimuli. The BCI decision-making process to discern a user's intent is error-prone due to noisy electro-physiological data, representing a classic problem of efficiently transmitting and receiving information via a noisy communication channel [31]. Consequently, we model the P300 speller as a noisy communication system. Within this framework, we maximize the spelling rate and the character selection accuracy by using a principled method that exploits coding theory to design the stimulus presentation paradigm.

2.1 The P300 speller as a noisy communication system with memory

The use of the P300 speller can be represented as a communication process through a noisy channel [32], as illustrated in figure 1. A user communicates an intended message or target character, C^* , from a set of M possible choices, $\{C_m\}_{m=1}^M$. A potential target character is encoded by its presentation pattern, $X_1^T = [x_1, x_2, \dots, x_T]$, which represents a binary codeword where each bit indicates the presence ($x_t = 1$) or absence ($x_t = 0$) of that character in a flash group, $\mathcal{F}_t$, at time index t . The Hamming weight of a codeword, $w^H(X_1^T)$, is the number of non-zero elements in a codeword and represents the target character frequency of the codeword.

During the character selection process, a series of flash groups, $\mathcal{F}_1^T = [\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_T]$, are presented to the user. Each flash group elicits a response depending on the type of stimulus event, where ideally a P300 ERP is elicited when the target character is presented, i.e. when $C^* \in \mathcal{F}_t$. These ERPs are embedded in noisy EEG data. Following each stimulus, the BCI analyzes the EEG data with a classifier to distinguish between target and non-target stimulus events, and generates a sequence of classifier scores, $Y_1^T = [y_1, y_2, \dots, y_T]$. A decoding algorithm is used to determine the target character by selecting the most likely sent codeword (or character), given the series of stimulus events presentations and scores, i.e. $P(X_1^T \text{ sent} | \mathcal{F}_1^T \text{ presented}, Y_1^T \text{ received})$.

The stimulus presentation paradigm can be represented as a binary codebook with a matrix, $\mathfrak{C} \in [0, 1]^{M \times T}$, where the rows of the codebook, $\mathfrak{C}(m, 1:T)$, are the mappings of each character to its codeword, and the columns of the codebook, $\mathfrak{C}(1:M, t)$, correspond to the flash groups. For most stimulus presentation paradigms, there is usually a base codebook design that is instantiated several times, with either the flash group order, e.g. [3], or the character-to-codeword assignments, e.g. [14, 32], randomized during each instantiation. We denote this base codebook as an (M, l) -code, and one instantiation of the base codebook as a *sequence*, where M is the number of codewords and l is the length of codewords in the base codebook. For r sequences, $1 \leq r \leq T/l$. For example, the base codebook of the RCP for an $R \times C$ grid is a $(R \times C, R+C)$ -code, where 1 sequence = $R + C$ flashes. For each sequence, the order of presentation of the row and column flash groups is randomized without replacement. For the 6×6 grid shown in figure 2(a), the corresponding (36, 12)-code for the RCP is shown in figure 2(b), with 12 flashes/sequence.

The ERP elicitation process is error prone due to various sources of noise, e.g. low ERP SNR, missing a target stimulus event, distractor stimuli, motion artifacts, etc. A binary input, x_t , indicating the absence or presence of the target character in a flash group is mapped to a classifier score random variable, y_t :

$$\text{If } C^* \notin \mathcal{F}_t, x_t = 0 \rightarrow y_t \sim p(y_t | H_0), \quad (1a)$$

$$\text{if } C^* \in \mathcal{F}_t, x_t = 1 \rightarrow y_t \sim p(y_t | H_1), \quad (1b)$$

where $p(y_t | H_0)$ and $p(y_t | H_1)$ are the classifier score likelihoods for non-target and target stimulus events, respectively. As a result, the transmission of multiple codewords may result in a similar series of classifier scores. The performance of the decoding algorithm depends not only on the ability of the BCI to discriminate between target and non-target stimulus events, but also on the discriminability between the possible series of classifier scores associated with each character's presentation.

During the decoding process, the probability distribution of the series of classifier scores associated with each character's presentation, $P(\mathbf{Y}_1^T | \mathbf{X}_1^T, \mathcal{F}_1^T)$, is determined by the character's flash pattern or codeword, conditioned on a specific character being the target character [37]. Characters with similar codewords are more likely to be confused with each other during the decoding process due to the increased correlation in the cumulative EEG responses associated with their respective presentations [?, 14]. One approach to facilitate the distinction of characters is to maximize the dissimilarities between codewords. The Hamming distance is a metric that quantifies the difference between two codewords by the number of positions in which they differ:

$$d^H(\mathbf{c}_i, \mathbf{c}_j) = \sum_{t=1}^T \mathbb{I}(c_{i,t} = c_{j,t}), \quad (2)$$

where $d^H(\mathbf{c}_i, \mathbf{c}_j)$ is the Hamming distance between codewords $\mathbf{c}_i$ and $\mathbf{c}_j$; and $c_{i,t}$ is the binary value at position t for codeword $\mathbf{c}_i$. For example, in the RCP, the pairwise Hamming distance between codewords is either 2 for characters that are in the same row or column, or otherwise 4. Consequently, erroneous BCI selections for the RCP are usually in the same row or column as the target character [14, 15], particularly those that are directly adjacent due to their spatial proximity to the target character. It should be noted that adding additional sequences increases the Hamming distance multiplicatively, e.g. presenting two sequences of the RCP doubles the Hamming distances between codewords.

A codebook can be designed to tolerate a certain number of transmission errors without compromising the decoding accuracy. A codebook can be defined by its *error-correcting capacity* [31],

$$e_b = \left\lfloor \frac{d_{\min}^H(\mathfrak{C}) - 1}{2} \right\rfloor, \quad (3)$$

where e_b is the maximum number of bit errors that are guaranteed to be corrected during the decoding process to correctly estimate the sent message; and $d_{\min}^H(\mathfrak{C})$ is the minimum pairwise Hamming distance in codebook $\mathfrak{C}$. For r sequences of the RCP, the minimum Hamming distance of the new codebook is $2r$. Consequently, with one sequence of RCP data, if the BCI system incorrectly selects the target character's row or column, the target character cannot correctly be identified unless at least one more sequence of data is collected.

However, a consequence of maximizing Hamming distances to increase error-correcting capacity is the selection of dense codewords, particularly those with repetitive character presentations, which can have a negative impact on BCI performance due to refractory effects [33, 34, 35, 36]. Refractory effects are indicators of memory in the neurophysiology response, i.e. preceding target stimulus events affect the ERP modulation process. In a channel with memory [39], the level of memory can be characterized as follows:

$$p(y_t | \mathbf{X}_1^t) = p(y_t | \mathbf{X}_{t-\delta}^t). \quad (4)$$

where δ is the number of preceding channel inputs that affect the current output. A channel is described as *memoryless* if the probability distribution of the current output only depends on the current input, i.e. $p(y_t | \mathbf{X}_1^t) = p(y_t | x_t)$. Focusing exclusively on increasing error-

correcting capacity would be suitable with a memoryless channel assumption, but this is not the case within the context of generating ERPs with an oddball paradigm.

It has been shown that the P300 ERP SNR and classification performance improves as the time interval between target character presentation increases [17, 19, 29, 40]. Let a target-to-target interval (TTI) of 1 and 3 be represented by the presentation patterns [...1 $\underline{1}$...] and [...100 $\underline{1}$...], respectively, where “ $\underline{1}$ ” denotes the target stimulus event under consideration. The relationship between classification performance and TTI is illustrated in figure 3, which shows the classifier likelihoods derived during system calibration and likelihoods for the target classifier scores obtained during testing, segregated by TTI. For shorter TTIs, the TTI-segregated pdf is more similar to the non-target classifier pdf. For example, for a target character presented twice in succession, i.e. a TTI of 1, the BCI system is more likely to infer the second presentation as a non-target stimulus event. The means of the TTI-segregated classifier scores increases as the TTI increases. For a TTI of 3 and above, the TTI-segregated pdf is more similar to the target pdf, as is desirable for more accurate target stimulus event classification.

We hypothesize that the relative proportion of shorter TTIs in a codebook determines the degree with which refractory effects negatively impact performance. In addition, for a TTI of 1, the effect of isolated double target character presentations, i.e. [...0110.....0110...], might be less detrimental to performance compared to repetitive series of target character presentations, e.g. [...001111100...]. The former scenario is likely to occur in the RCP, while the latter is a common characteristic of the previously proposed codebooks in [34, 35, 33] where performance was similar to, or worse than with the RCP.

A memoryless channel assumption or a channel with short-term memory is too simplistic to characterize the ERP elicitation process when exploiting an information theoretic-approach to design codebooks for the P300 speller. Consequently, we consider a noisy channel with longer term memory to preceding target stimulus events to account for refractory effects. In the next section, we describe our method to develop a codebook that maximizes performance while minimizing refractory effects.

2.2 Performance-based codebook development

Our overall goal is to develop a codebook such that a potential target character is selected with as few flash group presentations as possible to improve spelling rates, and the character choices are better distinguishable during the decoding process to improve selection accuracy. We define a combinatorial problem to select an (M, l) -code with the following objective function:

$$\underset{\mathcal{C} \in \{0, 1\}^{M \times l}}{\text{minimize}} \text{ EST}(\alpha_{\mathcal{C}}), \text{ subject to } A(\alpha_{\mathcal{C}}) \geq A_{\min}, \quad (5)$$

where EST is the expected stopping time; A is the selection accuracy; $A_{\min}$ is the minimum accuracy desired; and $\alpha_{\mathcal{C}}$ is a generic parameter that we define to quantify a user's performance level with a given codebook, $\mathcal{C}$. A user's performance level is determined by

the BCI system's ability to distinguish between EEG responses due to non-target and target stimulus event presentations, and can be affected by several factors such as the SNR, refractory effects, attention level, fatigue, etc. We hypothesize that the TTI statistics of a codebook also affects a user's performance level with a given codebook configuration depending on the relative proportion of short TTIs that contribute to refractory effects.

Selecting a codebook from a 2^I codeword search space requires a method to predict BCI performance with different codebook configurations. We also narrow down the 2^I search space to a smaller space of codewords with parameters that are tuned to positively affect performance. We denote the stimulus presentation paradigm developed with our method as the *performance-based paradigm* (PBP).

2.2.1 BCI performance prediction—We developed a method to predict BCI performance with a codebook given a user's classifier performance [37]. The performance prediction method is based on a probabilistic model of the target character estimation process of a generic ERP-based BCI, which is illustrated in figure 4. Every ERP-based BCI uses a defined function that quantifies how likely each character, $\{C_m\}_{m=1}^M$, is the target character after processing the responses to the stimulus events. For example, this function can be based on the cumulative moving average (CMA) of classifier scores [3], or probability values, e.g. [36]. We will denote a generic function used to quantify how likely each of the BCI characters is the target character as the *character cumulative response function* (CCRF), represented as $\{\Theta_m(t)\}_{m=1}^M$, at time index t for an M -choice BCI.

During a selection process, a user intends to select a target character, C^* . A stimulus event is selected based on a *stimulus selection rule* and is presented to the user. In a visual P300 speller, the presented flash groups are selected based on a given codebook, e.g. for the RCP, random selections of only row or column flash groups. Following each stimulus event presentation, the resulting classifier score after analyzing EEG data, y_b , is used to update $\{\Theta_m(t)\}_{m=1}^M$. For a given character, C_m , its CCRF value, $\Theta_m(t)$, is updated based on its presentation pattern, i.e. its codeword. For example, in the CMA algorithm, a character's CMA score is only updated when it is presented, i.e. if $C_m \in \mathcal{A}_t$.

A *stopping rule* is used to determine when to stop data collection, at a stopping time, t_s . In a static stopping algorithm, t_s is fixed. In a dynamic stopping algorithm, t_s is variable as data collection is stopped when the maximum CCRF value, $\Theta_{\max}(t)$, attains a threshold value, Θ_{thr} , or a data collection limit, $t_{\max}$, is reached. For example, for a DS algorithm with a CCRF based on probability values, data collection is stopped when any character's probability attains a threshold value. After data collection, the BCI makes a selection based on a *decision rule*, where the target character estimate, $\hat{C}^*$, is the character with the maximum CCRF value. The selected character, $\hat{C}^*$, can be used as feedback to inform the selection process for the next target character, e.g. via a language model [11]. In some BCI implementations, the BCI selection decision can be vetoed based on an auto-corrective mechanism [11, 41]. Our performance prediction method considers only the BCI selection performance based on the CCRF.

The CCRF values, $\{\Theta_m(t)\}_{m=1}^M$, represent a set of random variables and their predicted probability distributions can be estimated for a given time index. The evolution of $\{\Theta_m(t)\}_{m=1}^M$ depends on how well the classifier discriminates between non-target and target stimulus events, and also how discriminable the codewords are from each other. Consequently, the probability distributions of $\{\Theta_m(t)\}_{m=1}^M$ depend on the codebook configuration and the performance of a user's classifier. The probability distribution of the CCRF value of a character, $\Theta_m(t)$, depends on its codeword, conditioned on the target character during a selection process. Characters with similar flash patterns will have a higher degree of correlation of their CCRF values and will be more likely to be confused with each other during the decoding process, e.g. in the RCP, characters in the same row or column [14].

The performance of the target character estimation process can be determined by evaluating the predicted distributions of $\{\Theta_m(t)\}_{m=1}^M$ and analyzing the BCI outcomes upon implementing the algorithm's stopping and decision rules. Performance functions for accuracy and the EST can be derived from the stopping and decision rules, respectively. To account for a dynamic stopping (DS) criterion, determining the EST requires a weighted average over all possible stopping times:

$$\text{EST} = \sum_{t=1}^{t_{\max}} tP(t_s = t), = \sum_{t=1}^{t_{\max}-1} tP(\{\Theta_{\max}(t-1) < \Theta_{th}\} \cap \{\Theta_{\max}(t) \geq \Theta_{th}\}) + t_{\max}P(\Theta_{\max}(t_{\max}-1) \geq \Theta_{th}), \quad (6)$$

where $P(t_s = t)$ is the probability of stopping at time, $t_s = t$, where t_s can either be the time index when $\Theta_{\max}(t)$ attains the threshold, Θ_{th} , or when the data collection limit is reached at $t_{\max}$.

The target character is correctly selected if at the stopping time, the character with the maximum CCRF value corresponds to the target character, i.e. $\hat{C}^* = C^*$ if $\Theta_i(t_s) > \max_{j \neq i} \{\Theta_j(t_s)\} \Big| C_i = C^*$. Similar to the EST, the accuracy is determined by a weighted average over all possible stopping times, as well as over all possible character choices:

$$A_i = P(\hat{C}^* = C_i \mid C_i = C^*) = \sum_{t=1}^{t_{\max}} P\left(\left\{t_s = t\right\} \cap \left\{\max_{j \neq i} \{\Theta_j(t)\} < \Theta_i(t)\right\} \mid C_i = C^*\right), \quad (7a)$$

$$A = \sum_{i=1}^M A_i P(C_i = C^*), \quad (7b)$$

where A_i is the conditional probability of correctly selecting a character C_i if it is the target character; and $P(C_i = C^*)$ is the prior probability of C_i being the target character.

Random variable analysis [41, 42, 43] can be used to obtain the analytic solutions to (6)–(7). Alternatively, Monte Carlo simulations can be used to estimate the performance functions, if the solutions to (6)–(7) are not tractable for a given CCRF. A detailed analysis for deriving the BCI performance functions is presented in [37].

Our probabilistic-model approach to performance prediction is independent of user performance level and stimulus paradigm condition. This is because of (i) the formulation of the CCRF as a generic representation of the BCI algorithm implementation, and (ii) the use of a prediction algorithm that simultaneously predicts BCI outcomes with a given codebook configuration across possible user performance levels. In this study, we illustrate the utility of the performance prediction method to analyze codebook design with a specific CCRF, the Bayesian dynamic stopping algorithm, which is described in the next section.

2.2.2 Bayesian dynamic stopping—For the Bayesian dynamic stopping (DS) algorithm [36], a probability distribution is maintained over the character choices, $\{P_m(t)\}_{m=1}^M$. This probability distribution represents the Bayesian level of confidence that each character is the target character given data collection. With each new stimulus presentation, the classifier score is used to update $\{P_m(t)\}_{m=1}^M$ by Bayesian inference:

$$P_m(t) = \frac{p_m(t)P_m(t-1)}{\sum_{j=1}^M p_j(t)P_j(t-1)}, \quad (8a)$$

$$p_m(t) = \begin{cases} p(y_t|H_0), & \text{if } C_m \notin \mathcal{F}_t \\ p(y_t|H_1), & \text{if } C_m \in \mathcal{F}_t \end{cases}, \quad (8b)$$

where $P_m(t-1)$ and $P_m(t)$ are the prior and posterior character probabilities, respectively; $p_m(t)$ is the character likelihood, assigned based on whether C_m is present in the current flash group, $\mathcal{F}_t$; and $p(y_t|H_0)$ and $p(y_t|H_1)$ are the classifier likelihood probability density functions (pdfs) for the non-target and the target classifier scores, respectively. Data collection is stopped when a character's probability attains a threshold probability, P_{thr} , within the data collection limit. After data collection, the character with the maximum probability is selected as the target character estimate.

The normalizing term in the update equation (8a) makes the analysis of the posterior probability distribution at each time step complex. However, due to the binary choice in the character likelihood assignments in this multi-hypothesis test (8b) and the existence of a Bayesian equivalent for the likelihood ratio test [43], the Bayesian probability can be replaced with a simple function based on the likelihood ratio of the classifier scores, $p(y|H_1)p(y|H_0)$. A detailed analysis of this domain substitution is provided in [37]. Instead of the Bayesian probability, we evaluate the cumulative log-likelihood ratio for each character:

$$L_m(t) = \sum_{\tau=1}^t l_m(\tau), \quad (9a)$$

$$l_m(\tau) = \begin{cases} \ln\left(\frac{p(y_\tau|H_1)}{p(y_\tau|H_0)}\right), & \text{if } C_m \in \mathcal{F}_\tau, \\ 0, & \text{if } C_m \notin \mathcal{F}_\tau \end{cases}, \quad (9b)$$

where $L_m(t)$ and $l_m(\tau)$ are the character cumulative and observational log-likelihood ratios, respectively. The pdf of $L_m(t)$ can be derived based on the codeword for the character C_m conditioned on the target character during a selection process. Since $L_m(t)$ is a random sum of $l_m(\tau)$, its pdf can be determined via recursive convolutions of the pdfs of $l_m(\tau)$.

Under a normality assumption, the detectability index is a distance measure that quantifies the discriminability between two pdfs, p_1 and p_0 , by their separation and spread [44]. To account for unequal variances, we use the definition provided in [46]:

$$d = \frac{\mu_1 - \mu_0}{\sqrt{0.5(\sigma_1^2 + \sigma_0^2)}} \quad (10)$$

where d is the detectability index; and (μ_i, σ_i^2) are the mean and variance parameters of p_i .

The log-likelihood ratio is parameterized by d :

$$l_m(\tau) \sim \begin{cases} \mathcal{N}\left(\frac{d^2}{2}, d^2\right), & \text{if } C^* \in \mathcal{F}_\tau \\ \mathcal{N}\left(-\frac{d^2}{2}, d^2\right), & \text{if } C^* \notin \mathcal{F}_\tau \end{cases}. \quad (11)$$

A user's performance level can be quantified by the detectability index between the target and non-target classifier scores. We parameterize the performance functions of the Bayesian

DS algorithm with a user's classifier detectability index, which allows us to conveniently explore performance bounds and visualize the performance functions in a linear space.

2.2.3 Performance-based parameters to restrict search space—The search space for selecting a subset of codewords is exponentially large; for binary codewords of length L , it consists of 2^L codewords. We consider a smaller search space of codewords based on parameters that are tuned towards enhancing the ERP elicitation process and maximizing BCI performance. We restrict our search space based on the following parameters: the minimum Hamming distance, the minimum TTI and the codeword Hamming weights.

The minimum Hamming distance of a codebook provides a measure of how dissimilar the codewords are: the more dissimilar the codewords, the more discriminable the character choices are during the decoding process (3). However, maximizing Hamming distances favors the selection of codewords with high Hamming weights, hence shorter TTIs. Considering the limitations of the ERP elicitation process, this results in increased refractory effects. Refractory effects can be minimized by penalizing codewords with shorter TTIs: we achieve this by imposing a minimum TTI for codewords. The codeword Hamming weight directly affects the TTI. Sparser codewords increase the likelihood of generating higher target classifier scores due to longer TTIs. However, longer TTIs increase the amount of time necessary to make a BCI decision. A trade-off between minimizing lower target classifier scores due to shorter TTIs and obtaining higher classifier scores for target characters with more presentations is necessary.

It should be noted that refractory effects reflect the non-linearities in the EEG responses based on the timing between target stimulus events; specifically the increase in P300 ERP SNR as the time interval between target stimulus events increases. As a measure of refractory effects, we use the TTI. This differs from a *refractory period* [47], for which a recovery time period necessary for ERP elicitation in an oddball paradigm is defined (analogous to a relative or absolute refractory period within the context of neuronal excitation). We use the former since the latter is somewhat undefined and may be dependent on the task.

The TTI depends on the stimulus presentation rate as determined by the stimulus duration and inter-stimulus interval (ISI). To determine an appropriate minimum TTI based on our stimulus presentation parameters, we performed classification analysis, similar to that in figure 3, for different TTI conditions. We analyzed data from a study that implemented three stimulus presentation paradigms [48]: RCP, random (RndP) and checkerboard (CBP). In the RndP, the characters in the flash groups are randomly generated, with the condition that within each sequence, each character is presented at least once before any other characters are presented again. The CBP [15] is a special case of the RndP where a minimum TTI is imposed and spatial restrictions with respect to a grid layout are placed on the composition of characters in a flash group. In all three paradigms, both the stimulus duration and ISI were set to 62.5 ms.

The TTI distribution of the three stimulus presentation paradigms are shown in figure 5. Due to a low frequency of some TTIs in certain stimulus paradigms, features were pooled across

calibration and test runs of a participant's BCI session and a 10-fold cross validation was performed on the pooled data. The classifier scores obtained from the test set were segregated by TTI, and kernel density estimation was used to estimate pdfs from a histogram of the TTI-segregated classifier scores. The Kullback-Leibler divergence (KLD) [49] provides a measure of the similarity between two pdfs, where lower KLDs indicate greater similarity between two pdfs (see appendix A). Data from TTI values with a low number of samples were excluded from the KLD analysis due to potential distortions in the shape of the estimated TTI-segregated pdfs, such as the pdfs for TTIs of 19 and 21 in figure 3, which are bimodal. Figure 6 shows the KLD between the TTI-segregated pdf estimated from target classifier scores obtained from the test set and the pdf of the target classifier likelihood estimated from the training set. The mean results across participants are shown, as well as standard deviations.

The overall performance of the BCI selection process depends on the performance of a classifier that is trained with target features that are aggregated into a single label, irrespective of TTI. The relationship between P300 SNR and TTI is reflected during classifier training, such that the average classifier score increases as TTI increases. Consequently, the KLD decreases and then starts to increase with increasing TTI because the TTI-segregated pdf shifts from left to right with respect to the aggregate target classifier pdf, as illustrated in figure 3 (see appendix A for details on the shape of the KLD function). The TTI-segregated pdfs of classifier scores resulting from shorter TTIs are noticeably dissimilar from the target classifier pdf, to a lesser degree in the CBP where a minimum TTI is imposed, compared to the RCP and the RndP. However, shorter TTIs represent a relatively large proportion of TTIs in the RCP compared to the RndP (see figure 5), which increases the likelihood of misclassifying target stimulus events with shorter TTIs in the former, compared to the latter.

The KLD is a useful visual measure for determining suitable TTI that substantially reduces refractory effects; however, it is not necessarily the case that selecting the TTI that minimizes the KLD function will optimize codebook performance. From our performance prediction analysis [37], given the same class conditional distribution of classifier scores, a codebook with a larger proportion of longer TTIs will result in increased ESTs. For example, selecting a TTI of 12 results in the greatest similarity between the TTI-specific and aggregate pdfs for the CBP; however, a TTI of this duration would substantially increase the EST. This would likely lead to a speller that while potentially more accurate, would be less time-efficient due to the much slower spelling speeds. Thus, it is not necessarily beneficial to select a paradigm-specific minimum TTI. We select a low TTI value that achieves a balance between minimizing refractory effects, maximizing target classifier scores and minimizing the EST. From this analysis, we estimated that a suitable minimum TTI that achieves all three aims was a TTI of 3 because of the substantial reduction in the KLD from TTI values of 1 and 2 in the RCP case, compared to the rate of change in the KLD at higher TTI values. Based on the stimulus timing parameters, this corresponds to a minimum time interval of 375 ms between a particular character's presentation.

2.2.4 Codebook development algorithm—It is necessary to estimate how a user's performance level, $\alpha\mathcal{C}$, changes with a given codebook configuration, as this affects the

amount of data collection necessary to achieve a certain accuracy level [9]. A user's performance level can shift significantly from one codebook to another due to several psycho-physiological factors, which include the degree of refractory effects based on the relative proportion of short TTIs in a codebook [33, 34, 35]. Without a good model of the ERP elicitation process, estimating this performance level change across all possible codebook configurations requires empirical data collection, which is infeasible.

In our current approach, we restrict our codebook search space to minimize refractory effects by imposing a minimum TTI. By minimizing refractory effects, we hypothesize that the ERP elicitation process is more or less rendered memoryless. We believe this is a fair assumption given evidence from previous studies that classification performance is similar for longer TTIs [17, 33]. This assumption simplifies our performance prediction analysis as it allows us to predict performance with a user's performance level that does not change significantly within this subspace.

Several approaches can be used to solve the combinatorial problem of selecting an (M, l) -code from a space of $M \times l$ binary matrices. In this study, we used a greedy search to iteratively build a codebook from the restricted search space, by adding a new codeword to a partially-filled codebook such that the objective function (5) is minimized with respect to the other codewords. Since we assume a user's performance level is fixed, minimizing the EST to achieve a certain accuracy level with a DS algorithm is equivalent to maximizing accuracy with a static stopping algorithm, given the same data collection limit in both stopping criteria [38]. Consequently, we achieve the same objective defined in (5) by maximizing the predicted accuracy.

Algorithm 1

Pseudo-code for performance-based codebook development

Input: M = codebook size; l = codeword length; ω = Hamming weights; $d_{\min}^H$ = minimum Hamming distance; $\text{TTI}_{\min}$ = minimum target-to-target interval, d = detectability index

Output: $\mathcal{C}_{\text{new}}$ = New performance-based codebook

- 1 **Function** pbpCodebook($M, l, \omega, d_{\min}^H, \text{TTI}_{\min}, d$)
- 2 $\mathcal{C}^{2^l \times l}$ = matrix of all 2^l l -bit codewords.
- 3 $\mathcal{C}^r = \mathcal{C}^{2^l \times l} \xrightarrow{\text{remove}} c_i \in \{w^H(c_i) \notin \omega, \text{TTI}(c_i) < \text{TTI}_{\min} \text{ and } d^H(c_i, c_j) < d_{\min}^H\}, c_i \in \mathcal{C}^{2^l \times l}$
- 4 $\mathcal{C}_{\text{new}} = \arg \max_{c_i} \sum_{j=1}^{|\mathcal{C}^r|-1} d^H(c_i, c_j), c_i \neq c_j, \{c_i, c_j\} \in \mathcal{C}^r$
- 5 $\mathcal{C}^r \xrightarrow{\text{remove}} \mathcal{C}_{\text{new}}$
- 6 **while** $|\mathcal{C}_{\text{new}}| < M$ **do**
- 7 **for** $k = 1 : |\mathcal{C}^r|$ **do**
- 8 $c_k = \mathcal{C}^r(k, 1 : l)$
- 9 $\mathcal{C}_{\text{temp}} = \mathcal{C}_{\text{new}} \xleftarrow{\text{add}} c_k$
- 10 $A_{\text{temp}}(k) = \text{predictedAccuracy}(\mathcal{C}_{\text{temp}}, d)$ ▷ (see
- 11 **end**
- 12 $a = \arg \max_k \{A_{\text{temp}}(k)\}_{k=1}^{|\mathcal{C}^r|}$
- 13 $c_{\text{new}} = \mathcal{C}^r(a, 1 : l)$
- 14 $\mathcal{C}_{\text{new}} = \mathcal{C}_{\text{new}} \xleftarrow{\text{add}} c_{\text{new}}$
- 15 $\mathcal{C}^r = \mathcal{C}^r \xrightarrow{\text{remove}} c_{\text{new}}$
- 16 **end**
- 17 **return** $\mathcal{C}_{\text{new}}$

Notes: $X \left\{ \xleftarrow{\text{add}} / \xrightarrow{\text{remove}} \right\} x$: add/remove x to/from X ; $\text{TTI}(c)$ = target-to-target intervals in codeword c .

Algorithm 1 outlines pseudo-code to iteratively construct a codebook based on a set of performance-based parameters. We constructed a codebook configuration of the PBP using this algorithm and tested online performance in comparison with the RCP in a healthy participant study.

3 Methods**3.1 Participants**

Twenty healthy participants were recruited from the student and work population at Duke University for a study approved by the university's Institutional Review Board. All participants gave informed consent prior to participating in this study. Participants were numbered in the order that they were recruited.

3.2 P300 speller task

Each participant underwent one BCI experiment session, which lasted between 1.5 to 2 hours, including breaks. A BCI session consisted of copy-spelling tasks using the P300 speller. In a copy-spelling task, the user is instructed by the BCI as to the character on which to focus. The word choices were randomly selected from a corpus obtained from the English Lexicon Project [50]. The BCI session was implemented in two blocks, where a block consisted of a calibration run and test run for each stimulus presentation paradigm condition. The block order was counterbalanced across participants to avoid order effects.

The calibration run involved data collection from copy-spelling five 6-letter words, with no classifier use or BCI feedback. Labeled data obtained from the calibration run were used to train user-specific classifier weight vector and likelihood functions. In the test run, using the trained classifier parameters, participants performed copy-spelling tasks of eight 6-letter words with feedback and no error correction, except for participants 1 and 3 with five 6-letter words. The Bayesian DS algorithm was used for character selection in the test run, with a probability threshold $P_{th} = 0.9$ and a data collection limit of $t_{max} = 72$ stimulus flashes.

3.3 Signal acquisition and processing

The open source BCI2000 software [51] was used to implement the P300 speller. EEG signals were acquired using 16-channel wet electrode caps (Electro-Cap) and relayed to a computer via g.USBamp (Guger Technologies) biosignal amplifiers. The left and right mastoids were used as ground and reference electrodes, respectively. Data collected from electrodes {Fz, Cz, P3, Pz, P4, PO7, PO8, Oz} were used for signal processing. EEG signals were sampled at 256 samples/s and filtered between 0.5–30 Hz. A time window of 800 ms of EEG response from stimulus onset was used to extract features for classification training or online evaluation.

Feature vectors, $\vec{f} \in \mathbb{R}^{1 \times 120}$, were obtained by down-sampling to around 13 Hz via bin-averaging [51]. A stepwise linear discriminant analysis (SWLDA) classifier was used for classification. Feature vectors obtained from data from the calibration run were labeled according to whether they were extracted following non-target ($\gamma = 0$) and target ($\gamma = 1$) character presentations. The training dataset of labeled feature vectors, $\mathcal{T} = \{(\vec{f}_1, \gamma_1), \dots, (\vec{f}_n, \gamma_n)\}$ was used to train the SWLDA classifier (using MATLAB[®], The MathWorks Inc., Natick, MA), with $p\text{-enter} = 0.1$ and $p\text{-remove} = 0.15$, to obtain a weight vector, $\vec{w} \in \mathbb{R}^{1 \times 120}$. Classifier scores from the training data were used to train the classifier likelihoods via kernel density estimation, to be used in the probability update step of the Bayesian DS algorithm. In the test run, a classifier score is obtained from a product of the classifier weight vector and feature vector extracted from EEG data, $y_t = \vec{w} \vec{f}_t^T$. Classifier scores from the test data were used to compute detectability indices for performance prediction analysis.

3.4 Stimulus presentation paradigms

The RCP and a configuration of the performance-based paradigm (PBP) were implemented based on the 6×6 speller grid shown in figure 2(a). The RCP was chosen as the baseline for comparison because it is the predominantly used stimulus presentation paradigm in the literature [13]. The codebook for the RCP consisted of the rows and columns of the grid, presented in a random order without replacement [3].

The method outlined in algorithm 1 was used to generate the PBP. A (36, 24)-code was generated to allow for a larger degree of freedom in selecting codewords with a more varied proportion of TTIs. Since performance is compared to the RCP, the number of codebook realizations for the RCP was doubled such that it was also a (36, 24)-code. For a RCP with a (36, 24)-code, the performance-based parameters are $d_{\min}^H = 4$, $w^H(c) = 1/6$, and $\text{TTI}_{\min} = 1$.

An example codebook for the RCP used in this study is shown in 7(a). Assuming a uniform distribution over characters, we used algorithm 1 to perform an iterative search over performance-based parameter values, $d_{\min}^H > 4$, $w^H(c) > 1/6$ and $\text{TTI}_{\min} = 3$. Additional consideration was taken to diversify the TTI statistics of codewords. Multiple codebook configurations were constructed using several parameter sets and a final configuration that minimized the ratio $\frac{\text{EST}}{A}$ over a wide range of d values was selected.

The (36, 24)-code configuration for the PBP used in this study is shown in figure 7(b). The axis of randomization of this codebook is the character-to-codeword assignments, in order to preserve the TTI characteristics of the codebook. The error-correcting capacity of the PBP codebook ($d_{\min}^H = 6$) is higher than the RCP ($d_{\min}^H = 4$). The PBP is able to present more of the character choices in a fewer amount of flashes. It can be observed that character presentation frequency is higher in the PBP codebook ($w^H(c) = 1/4$) compared in RCP codebook ($w^H(c) = 1/6$), hence larger proportion of shorter TTIs in the former (PBP: mean TTI ≈ 4) compared to the latter (RCP: mean TTI ≈ 5.7). The flash group sizes in the PBP (8–11 characters) are also larger than in the RCP (6 characters). It can also be observed in figure 7(a) that double flashes, i.e. TTI of 1, are likely occur in the RCP.

3.5 Performance measures

The performance measures were estimated based on data from the test run and include: accuracy, the EST, the mean character selection time, bit rate [53] and detectability indices. The accuracy, A , is the percentage of characters spelled correctly during testing. The detectability index was calculated from the classifier scores according to (10). The mean character selection time is the average time to select each character, which includes the time pause between character selections:

$$\bar{T} = \frac{1}{N} \sum_{n=1}^{N_s} 3.5 + t_n(\text{ISI} + \text{FD}) \text{ seconds}, \quad (12)$$

where $\bar{T}$ is the mean character selection time for character, N_s is the total number of spelled characters; 3.5 seconds is the time pause between character selections; t_n is the number of flashes recorded prior to terminating data collection for the n^{th} character; and ISI and FD are the inter-stimulus interval and flash duration, respectively, both set to 62.5 ms.

Bit rate is a communication measure that takes into account accuracy, the mean character selection time and the number of choices,

$$B = \log_2 M + A \log_2 A + (1 - A) \log_2 \left(\frac{1 - A}{M - 1} \right) \text{ bits}, \quad (13)$$

$$BR = \frac{B}{\bar{T}} \times 60 \text{ bits/min}, \quad (14)$$

where B is the number of transmitted bits per character selection; M is the number of possible character selections in the speller; and BR is the bit rate.

Data from all of the recruited participants were used for performance analysis. Statistical significance was tested using the Wilcoxon signed-ranked test, ($p < 0.05$).

4 Results

The participant-specific results are shown in figures 8 and are summarized in table 1. Figure 8(a) shows the mean character selection time, after taking into account the flash duration, ISI and the 3.5 seconds time pause between character selections. The mean character selection time was significantly reduced with the PBP, $p < 10^{-4}$. Figure 8(b) shows participant accuracy. Despite a significant reduction in the amount of data collection, for most participants, accuracy was either similar across paradigms or increased with the PBP. A significant improvement in accuracy was observed with the PBP, $p < 10^{-2}$. Figure 8(c) shows participant communication rates, in bits per minute (bits/min). Due to similar or improved accuracy while reducing the amount of time per character selection, a significant improvement was observed in bit rate with the PBP, $p < 10^{-3}$.

When developing the codebook for the PBP, we assumed that a user's performance level didn't change significantly between codebooks, in order to obtain the necessary performance improvements from the RCP. Participant-specific d values are shown figure 8(d). Participant d values were more or less similar, with no significant differences between the two stimulus presentation paradigms, $p = 0.15$. Figure 9 compares the online performances to those predicted by the d values of their respective test data. The online results exhibit a high degree of variability due to estimating the performance measures from a low number of samples (30 or 48 characters). Nonetheless, the performance trends of the current results are consistent with the offline performance predictions.

5 Discussion

5.1 Analysis of codebook performance

We obtained significant performance improvements with the PBP, and substantial improvements occur in different ranges of user performance levels for accuracy and the EST. With a DS algorithm, the rate of algorithm convergence to the decision threshold is determined by the discriminability between target and non-target classifier scores and the average interval between potential target character presentations. A codebook where a potential target character is presented more often due to shorter TTIs is more likely to achieve faster convergence with a DS algorithm. Consistent with our predictions, a decrease in the EST was observed with the PBP across most participants, compared to the RCP.

A higher dissimilarity between codewords increases the discriminability of the character choices during the decoding process and improves decoding accuracy. The codewords in the PBP are more dissimilar than in the RCP. Hence, a significant improvement in accuracy is observed with the PBP in most of the lower-level performers with the RCP. The data collection limit has negligible effects on the maximum attainable accuracy at higher d values because the DS algorithm is more likely to converge well below this limit at these d values; this is illustrated in figure 9 by the predicted performances for both codebooks at $t_{\max} \approx \infty$ and $t_{\max} = 72$. As a result, similar accuracies are observed in the higher level performers. The joint consideration of accuracy and spelling time is key for successful BCI use. While accuracy is only slightly improved for higher level performers, the new codebook is able to improve spelling speed, resulting in improved overall performance for all levels of performers.

Our approach to stimulus paradigm design was based on an optimization problem to improve performance given certain physiological limitations of our communication system. We imposed a minimum TTI to mitigate refractory effects and assumed a noisy memoryless channel model during codebook optimization. We chose a set of performance-based parameters to narrow down our search space based on the stimulus presentation parameters of our BCI application. A different set of parameters or objective function may be chosen to suit the BCI design needs under consideration. Within a noisy channel framework, other channel models that explicitly model refractory effects during the ERP elicitation process can be used to optimize codebook design. A recent extension of this work proposes an alternative theoretical framework based on finite-state channels with memory to model and analyze the impact of refractory effects on BCI performance [54]. Preliminary results from simulations show the potential to use this framework to design codebooks that closely achieve channel capacity within the refractory effect limitations defined by a specific channel model.

It is important to note that our codebook design approach relies on the performance predictions reflecting online performance trends. We defined a codebook search space where we assumed that a user's performance level did not change significantly, which we achieved by minimizing refractory effects. Refractory effects can generally be controlled by imposing a minimum time interval between target stimulus presentations. However, there potentially exists a point of diminishing returns where other psycho-physiological effects that are not

accounted for, e.g. fatigue, increased cognitive load, user distractions, etc., may negate any potential performance improvements obtained with an optimized codebook design. For example, users might be more prone to experience visual fatigue or attentional blindness at high stimulus flash rates [55] or with codebook designs with a higher degree of adjacent distracting stimuli [15]. While these were not a concern in this study due the small flash group sizes in our current PBP configuration, visual fatigue effects may become significant with larger grid sizes where there is the potential to create larger flash group sizes with limited spatial spread.

5.2 Comparison with previous work

As with our study, other approaches impose a minimum TTI to mitigate refractory effects, e.g. [15, 27, 28, 29, 30], and these approaches usually select stimulus flash patterns using on a set of heuristics. Polprasert *et al.* [30] modified the RCP stimulus presentation by using a circular delay approach to impose a minimum TTI, while Jin *et al.* [29] determined combinations of character flash patterns using binomial coefficients [29]. Other approaches also consider performance-guided constraints to select stimulus presentation patterns based on criteria where potential performance improvements are inferred during the design process. The CBP developed by Townsend *et al.* (2010) [15] uses a checkerboard pattern on a grid lay-out to impose spatial restrictions on the composition of flash groups, in order to minimize adjacency distraction errors. Their framework was extended to the temporal domain in [27, 28], which involved imposing restrictions on how often characters can share the same flash group. In addition, [28] also addresses some technical limitations of stimulus timing in the BCI2000 open-source software platform [51] that is widely used in most BCI studies, including in this study.

A main limitation of these previous heuristic-based methods is that these approaches do not provide a flexible framework that allows for adaptability of paradigm conditions. These previous methods may not necessarily be optimized for a dynamics stopping algorithm, where shorter TTIs achieve faster algorithm convergence. Codebook design methods that favor the selection of sparser codewords, e.g. [15, 29], usually lead to increases in the EST due to the longer TTIs. Also, heuristic-based approaches may not necessarily generalize when there are a limited number of BCI choices. For example, the minimum TTI imposed with the CBP increases as the grid size of the interface increases. While most reasonable grid sizes might result in a minimum TTI that is potentially sufficient to minimize refractory effects, the lower bound for the EST will always be limited by the minimum TTI imposed by a particular grid size. Some previous design methods impose restrictions on flash group composition, e.g. [27, 28], which can be problematic as the number of BCI choices reduces. A probabilistic approach allows us to explore the performance bounds of a BCI algorithm with different codebook configurations across a wide range of performance levels. By evaluating codebooks by their predicted performances, this provides a more principled way to accommodate dynamic stopping algorithms, as well as generalize across various grid size options without imposing restrictions on flash group composition. A different grid size or set of stimulus timing parameters will likely result in a different optimized codebook configuration using our approach.

Similar to this study, several other approaches to P300 speller codebook design also focus on developing error-correcting codes, such as the studies in [33, 34, 35, 36]. The approaches in [33, 34, 35] focused primarily on maximizing Hamming distances. Hill *et al.* [33] also proposed an optimized codebook design based on minimizing a set of error-loss functions. Verhoeven *et al.* [36] employed additional heuristics to reduce the proportion of shorter TTIs and also considered cosmetic aspects of the grid to minimize adjacency distraction errors. The benefit of error-correcting codes is the use of an objective criterion to evaluate performance during the codebook design process: the more codewords are dissimilar, the better the ability to distinguish between the character choices during the decoding process.

However, the previous error-correcting codes proposed for the P300 speller did not obtain the expected performance improvements from the RCP, possibly due to underestimating the adverse impact of refractory effects at the given stimulus presentation rates. These previous methods favored the selection of dense codebooks with predominantly shorter TTIs. For example, for a (36, 24)-code, the D10 codebook proposed in [33] has a $d_{\min}^H = 10$, with 43% TTI of 1 and 24% TTI of 2, compared to the RCP, with $d_{\min}^H = 4$, 11% TTI of 1 and 11% TTI of 2. Despite its increased error-correcting capacity (3), the D10 codebook performed significantly worse than the RCP. A solution was proposed in [33] to account for refractory effects by considering a channel with one level of memory during the decoding process, i.e. $p(y_t | \mathbf{X}_1^t) = p(y_t | x_{t-1}, x_t)$. This approach was shown to significantly improve performance of a Hadamard (HAD) codebook proposed in [34] compared to a memoryless decoding strategy, i.e. $p(y_t | \mathbf{X}_1^t) = p(y_t | x_t)$. However, the RCP still significantly outperformed the HAD codebook (for a (36, 24)-code, $d_{\min}^H = 12$) despite its higher error-correcting capacity, which suggests that a short-term memory consideration may be insufficient.

It may be necessary to fully characterize the impact of refractory effects on performance for longer series of target character presentations to obtain substantial benefits with a memory-based decoding strategy; however, this approach is likely to increase the complexity of the decoder. For example, the response to a current target stimulus event may depend on the number of preceding target stimuli, e.g. [...01 $\underline{1}$...], [...011 $\underline{1}$...], [...0111 $\underline{1}$...], etc., where $\underline{1}$ is the current event under consideration. While we specify a $d_{\min}^H$ parameter to guide our codebook selection, we do not rely predominantly on a codebook's error-correcting capacity as an indicator of performance. We account for refractory effect limitations by imposing a minimum TTI; this also allows us to optimize our codebook design with the simple assumption of a memoryless channel.

There is the potential to obtain additional performance improvements by using salient elements in the visual interface during stimulus presentation, to elicit additional ERPs that enhance performance [24, 25, 26]. The use of salient stimuli was shown to compensate for the negative impact of refractory effects on performance with denser codebooks [33, 35, 56], which sometimes resulted in comparable performance with the RCP. However, the improved accuracies observed with a static stopping algorithm may not necessarily hold when using a DS algorithm, if higher bit classification errors are observed with denser codebooks [35].

Without the use of salient stimuli, we obtained significant performance improvements using our codebook configuration with a DS algorithm. Investigation of salient stimuli with a DS algorithm may provide insight to further improve performance with our codebook design.

5.3 Future work

The method to optimize codebook design for the P300 speller will be tested in a target end-user population. It should be noted that performance trends from healthy participant studies may not always translate to individuals with severe neuromuscular limitations. Potential factors we anticipate to affect codebook design in a target population include differences in psycho-physiological responses to stimuli. For example, target BCI end users are less tolerant of high stimulus presentation rates: we typically use 4 Hz and 8 Hz stimulus presentation rates in our studies with target BCI end users [9] and healthy users [8, 37], respectively. Additional consideration in the choice of stimulus parameters is required for optimizing codebook design for a target BCI user population.

We optimized the stimulus presentation paradigm design assuming a static codebook: even though the character-to-codeword assignments can be randomized, the base codebook design of the stimulus presentation paradigm does not change. Similar to a dynamic stopping algorithm for data collection, performance improvements can potentially be obtained by creating flash groups dynamically in order to maximize the information content of future flash group presentations based on the current BCI data. We are currently exploring using performance prediction analysis for an adaptive codebook design. Also, language information can be exploited during the BCI selection process to improve performance [11]. In this study, we did not exploit language information either during codebook development or in the online implementation of the Bayesian DS algorithm [8] to avoid introducing a language model as a confounding factor when demonstrating the utility of our method. Future work includes an analysis of the impact of language models on codebook performance.

6 Conclusions

We have developed a principled approach to design stimulus presentation patterns for the P300 speller based on a probabilistic method to predict BCI performance. Within a noisy channel framework, we design codebooks to achieve a balance between increasing error-correcting capacity, minimizing refractory effects and maximizing the spelling rate, by taking into account the transmission dynamics of the communication channel. In an online experiment with healthy participants, we obtained statistically significant improvements in accuracies and spelling rates across a wide range of user performance levels. The proposed stimulus paradigm design method will be tested in a target BCI population.

Acknowledgments

This research project was funded by the NIH/NIDCD under grant number R33 DC010470 and the Kristina M. Johnson Fellowship. The authors would like to thank the participants who volunteered their time for this study. The authors would also like to thank the two anonymous reviewers for their respective comments.

Appendix A Kullback-Leibler divergence

The Kullback-Leibler divergence (KLD) [48] is the non-symmetric difference between two probability distributions over the same variable. It is computed accordingly:

$$D_{KL}(P \parallel Q) = \sum_i P(i) \log \left(\frac{P(i)}{Q(i)} \right) \quad (15)$$

$$D_{KL}(p \parallel q) = \int_{-\infty}^{\infty} p(i) \log \left(\frac{p(i)}{q(i)} \right) dx \quad (16)$$

where $D_{KL}(P \parallel Q)$ is the KLD between probability distributions P and Q , and (15) and (16) represent the formulas for discrete and continuous random variables, respectively.

The KLD is non-negative, equals to zero if and only if $P = Q$. The KLD provides a non-directional visual measure of the degree of overlap between two probability distributions: the higher the degree of overlap between two probability distributions, the smaller the KLD. For example, consider the KLD between a standard normal distribution, f_0 , and a horizontal transformation, q . Figure 10 shows the KLD as a function of the horizontal shift, $D_{KL}(q \parallel f_0)$. The function is convex, hence the KLD measure provides no information of the direction of the horizontal shift.

References

1. Wolpaw, JR., Wolpaw, EW. Brain-computer interfaces: principles and practice New York: Oxford University Press; 2012
2. Moghimi S, Kushki A, Guerguerian AM, Chau T. A review of EEG-based brain-computer interfaces as access pathways for individuals with severe disabilities. *Assistive Technology*. 2013; 25(2):99–110. [PubMed: 23923692]
3. Farwell LA, Donchin E. Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials. *Electroencephalography and Clinical Neurophysiology*. 1988; 70(6):510–523. [PubMed: 2461285]
4. Sutton S, Braren M, Zubin J, John ER. Evoked-potential correlates of stimulus uncertainty. *Science*. 1965; 150(3700):1187–8. [PubMed: 5852977]
5. Krusienski DJ, Sellers EW, Cabestaing F, Bayouth S, McFarland DJ, Vaughan TM, et al. A comparison of classification techniques for the P300 Speller. *Journal of Neural Engineering*. 2006; 3(4):299–305. [PubMed: 17124334]
6. Manyakov NV, Chumerin N, Combaz A, Van Hulle MM. Comparison of classification methods for P300 brain-computer interface on disabled subjects. *Computational Intelligence and Neuroscience*. 2011; 2011:519868. [PubMed: 21941530]
7. Aloise F, Schettini F, Aricò P, Salinari S, Babiloni F, Cincotti F. A comparison of classification techniques for a gaze-independent P300-based brain-computer interface. *Journal of Neural Engineering*. 2012; 9(4):045012. [PubMed: 22832242]
8. Mainsah BO, Colwell KA, Collins LM, Throckmorton CS. IEEE Transactions on Neural Systems and Rehabilitation Engineering. Utilizing a Language Model to Improve Online Dynamic Data Collection in P300 Spellers. 2014; 22(4):837–846.

9. Mainsah BO, Collins LM, Colwell KA, Sellers EW, Ryan DB, Caves K, et al. Increasing BCI communication rates with dynamic stopping towards more practical use: an ALS study. *Journal of Neural Engineering*. 2015; 12(1):016013. [PubMed: 25588137]
10. Schreuder M, Höhne J, Blankertz B, Haufe S, Dickhaus T, Tangermann M. Optimizing event-related potential based brain-computer interfaces: a systematic evaluation of dynamic stopping methods. *Journal of Neural Engineering*. 2013; 10(3)
11. Speier W, Arnold C, Pouratian N. Integrating language models into classifiers for BCI communication: a review. *Journal of Neural Engineering*. 2016; 13(3):031002. [PubMed: 27153565]
12. Orhan U, Hild KE, Erdogmus D, Roark B, Oken B, Fried-Oken M. RSVP keyboard: An EEG based typing interface. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2012:645–648.
13. Mak JN, McFarland DJ, Vaughan TM, McCane LM, Tsui PZ, Zeitlin DJ, et al. EEG correlates of P300-based brain-computer interface (BCI) performance in people with amyotrophic lateral sclerosis. *Journal of Neural Engineering*. 2012; 9(2):026014. [PubMed: 22350501]
14. Townsend G, LaPallo BK, Boulay CB, Krusienski DJ, Frye GE, Hauser CK, et al. A novel P300-based brain-computer interface stimulus presentation paradigm: moving beyond rows and columns. *Clinical Neurophysiology*. 2010; 121(7):1109–20. [PubMed: 20347387]
15. Citi L, Poli R, Cinel C. Exploiting P300 amplitude variations can improve classification accuracy in Donchin's BCI speller. *4th International IEEE/EMBS Conference on Neural Engineering*. 2009:478–481.
16. Martens SMM, Hill NJ, Farquhar J, Schölkopf B. Overlap and refractory effects in a brain-computer interface speller based on the visual P300 event-related potential. *Journal of Neural Engineering*. 2009; 6(2)
17. McFarland DJ, Sarnacki WA, Townsend G, Vaughan T, Wolpaw JR. The P300-based brain-computer interface (BCI): effects of stimulus rate. *Clinical Neurophysiology*. 2011; 122(4):731–737. [PubMed: 21067970]
18. Lu J, Speier W, Hu X, Pouratian N. The effects of stimulus timing features on P300 speller performance. *Clinical Neurophysiology*. 2013; 124(2):306–314. [PubMed: 22939456]
19. Sellers EW, Krusienski DJ, McFarland DJ, Vaughan TM, Wolpaw JR. A P300 event-related potential brain-computer interface (BCI): the effects of matrix size and inter stimulus interval on performance. *Biological Psychology*. 2006; 73(3):242–252. [PubMed: 16860920]
20. Takano K, Komatsu T, Hata N, Nakajima Y, Kansaku K. Visual stimuli for the P300 brain-computer interface: A comparison of white/gray and green/blue flicker matrices. *Clinical Neurophysiology*. 2009; 120(8):1562–1566. [PubMed: 19560965]
21. Treder MS, Schmidt NM, Blankertz B. Gaze-independent brain-computer interfaces based on covert attention and feature attention. *Journal of Neural Engineering*. 2011; 8(6)
22. Gavett S, Wygant Z, Amiri S, Fazel-Rezai R. Reducing human error in P300 speller paradigm for brain-computer interface. *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 2012:2869–2872.
23. Kaufmann T, Schulz SM, Grünzinger C, Kübler A. Flashing characters with famous faces improves ERP-based brain-computer interface performance. *Journal of Neural Engineering*. 2011; 8(5):056016. [PubMed: 21934188]
24. Kaufmann T, Schulz SM, Köblitz A, Renner G, Wessig C, Kübler A. Face stimuli effectively prevent brain-computer interface inefficiency in patients with neurodegenerative disease. *Clinical Neurophysiology*. 2013; 124(5):893–900. [PubMed: 23246415]
25. Jin J, Daly I, Zhang Y, Wang X, Cichocki A. An optimized ERP brain-computer interface based on facial expression changes. *Journal of Neural Engineering*. 2014; 11(3):036004. [PubMed: 24743165]
26. Townsend G, Shanahan J, Ryan DB, Sellers EW. A general P300 brain-computer interface presentation paradigm based on performance guided constraints. *Neuroscience Letters*. 2012; 531(2):63–68. [PubMed: 22960261]

27. Townsend G, Platsko V. Pushing the P300-based brain-computer interface beyond 100 bpm: extending performance guided constraints into the temporal domain. *Journal of Neural Engineering*. 2016; 13(2):026024. [PubMed: 26913648]
28. Jin J, Sellers EW, Wang X. Targeting an efficient target-to-target interval for P300 speller brain-computer interfaces. *Medical & Biological Engineering & Computing*. 2012; 50(3):289–296. [PubMed: 22350331]
29. Polprasert C, Kukieattikool P, Demeechai T, Ritcey JA, Siwamogsatham S. New stimulation pattern design to improve P300-based matrix speller performance at high flash rate. *Journal of Neural Engineering*. 2013; 10(3):036012. [PubMed: 23612883]
30. MacKay, DJC. *Information theory, inference and learning algorithms* Cambridge University Press; 2003
31. Cover, TM., Thomas, JA. *Elements of Information Theory* John Wiley & Sons; 2006
32. Hill, J., Farquhar, J., Martens, S., Biessmann, F., Schölkopf, B. *Advances in Neural Information Processing Systems* Curran Associates, Inc.; 2009 Effects of Stimulus Type and of Error-Correcting Code Design on BCI Speller Performance; 665–672
33. Martens S, Mooij JM, Hill NJ, Farquhar J, Schölkopf B. A graphical model framework for decoding in the visual ERP-based BCI speller. *Neural Computation*. 2011; 23(1):160–182. [PubMed: 20964540]
34. Geuze J, Farquhar JDR, Desain P. Dense codes at high speeds: varying stimulus properties to improve visual speller performance. *Journal of Neural Engineering*. 2012; 9(1):016009. [PubMed: 22248483]
35. Verhoeven T, Buteneers P, Wiersema JR, Dambre J, Kindermans P. Towards a symbiotic brain-computer interface: exploring the application-decoder interaction. *Journal of Neural Engineering*. 2015; 12(6):066027. [PubMed: 26580120]
36. Throckmorton CS, Colwell KA, Ryan DB, Sellers EW, Collins LM. Bayesian approach to dynamically controlling data collection in P300 spellers. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*. 2013; 21(3):508–17. [PubMed: 23529202]
37. Mainsah BO, Collins LM, Throckmorton CS. Using the detectability index to predict P300 speller performance. *Journal of Neural Engineering*. 2016; 13(6):066007. [PubMed: 27705956]
38. Gallager, RG. *Information Theory and Reliable Communication* New York, NY, USA: John Wiley & Sons, Inc; 1968
39. Gonsalvez CJ, Polich J. P300 amplitude is determined by target-to-target interval. *Psychophysiology*. 2002; 39(3):388–396. [PubMed: 12212658]
40. Chavarriaga R, Sobolewski A, Millán JdR. Errare machinale est: the use of error-related potentials in brain-machine interfaces. *Frontiers in Neuroscience*. 2014; 8:208. [PubMed: 25100937]
41. Trivedi, KS. *Probability & statistics with reliability, queuing and computer science applications* John Wiley & Sons; 2008
42. Clark CE. The greatest of a finite set of random variables. *Operations Research*. 1961; 9(2):145–162.
43. Wald A. Sequential Tests of Statistical Hypotheses. *The Annals of Mathematical Statistics*. 1945; 16(2):117–186.
44. Kay, SM. *Fundamentals of Statistical Signal Processing, Volume II: Detection Theory* Englewood Cliffs, NJ: Prentice-Hall PTR; 1993
45. Simpson AJ, Fitter MJ. What is the best index of detectability? *Psychological Bulletin*. 1973; 80(6):481–488.
46. Johannsen J, Röder B. Uni- and crossmodal refractory period effects of event-related potentials provide insights into the development of multisensory processing. *Frontiers in Human Neuroscience*. 2013; 8:552–552.
47. Throckmorton, CS., Ryan, DB., Hamner, B., Caves, K., Colwell, KA., et al. Towards Clinically Acceptable BCI Spellers: Preliminary Results for Different Stimulus Selection Patterns and Pattern Recognition Techniques; Fourth International BCI Meeting; Asilomar, California. 2010L-29
48. Kullback S, Leibler RA. On information and sufficiency. *The Annals of Mathematical Statistics*. 1951; 22(1):79–86.

49. Balota DA, Yap MJ, Hutchison KA, Cortese MJ, Kessler B, Loftis B, et al. The English Lexicon Project. *Behavior Research Methods*. 2007; 39(3):445–459. [PubMed: 17958156]
50. Schalk, G., Mellinger, JA. *Practical Guide to Brain-Computer Interfacing with BCI2000: General-Purpose Software for Brain-Computer Interface Research, Data Acquisition, Stimulus Presentation, and Brain Monitoring* Springer; 2010
51. Krusienski DJ, Sellers EW, McFarland DJ, Vaughan TM, Wolpaw JR. Toward enhanced P300 speller performance. *Journal of Neural Engineering*. 2008; 167(1):15–21.
52. McFarland DJ, Sarnacki WA, Wolpaw JR. Brain-computer interface (BCI) operation: Optimizing information transfer rates. *Biological Psychology*. 2003; 63(3):237–251. [PubMed: 12853169]
53. Mayya V, Mainsah BO, Reeves G. Modeling the P300-based brain-computer interface as a channel with memory. 54th Annual Allerton Conference on Communication, Control, and Computing. 2016:23–30.
54. Gao S, Wang Y, Gao X, Hong B. Visual and auditory brain-computer interfaces. *IEEE Transactions on Biomedical Engineering*. 2014; 61(5):1436–1447. [PubMed: 24759277]
55. Martens SMM, Leiva JM. A generative model approach for decoding in the visual event-related potential-based brain-computer interface speller. *Journal of Neural Engineering*. 2010; 7(2): 026003.

Figure 1.

Schematic of a communication system where a message is transmitted from one point to another via a noisy channel. The message, C^* , is encoded with a codeword, $X_1^T = [x_1, x_2, \dots, x_T]$. The codeword is transmitted through a noisy channel, which results in an output sequence, $Y_1^T = [y_1, y_2, \dots, y_T]$, at the receiver. The received sequence is used to estimate the sent message, $\hat{C}^*$, based on a decoding rule.

Figure 2.
(a) Example visual P300 speller interface with a 6×6 grid layout. To select a target character, a user focuses on that character in the grid while groups of characters are sequentially illuminated on the screen while the system analyzes EEG data in order to make a character selection. In this example, the last row is illuminated. (b) Corresponding base codebook for the row-column paradigm (RCP). Each row represents the codeword for a character. Each column in the codebook represents a flash group, with characters presented highlighted in white. For example, the highlighted row in the interface on the left is represented by flash group 6. The axis of randomization for this codebook (indicated in the red) is the order of presentation of the flash groups, to maintain the character groupings of rows and columns.

Figure 3.

Illustration of refractory effects on classification performance for a study participant in [36]. Refractory effects are quantified by the target-to-target interval (TTI). Each panel shows probability density functions (pdfs) of the classifier likelihoods for non-target (H_0^{train}) and target (H_1^{train}) responses obtained from the training data, and that of the target classifier scores segregated by TTI for the test data ($H_{\text{TTI}}^{\text{test}}$). The respective pdfs were obtained via kernel density estimation from a histogram of the classifier scores, y , where n denotes the number of observations used for estimation.

Figure 4.

Schematic of the target character estimation process of an ERP-based BCI. A user attends to a target character, C^* , from a set of choices, $\{C_m\}_{m=1}^M$. Based on a codebook configuration, the BCI presents a stimulus to a user, $\mathcal{F}_t$, which may elicit an ERP. The user's EEG response is scored with the system's classifier. The classifier score, y_t , is used to update a character cumulative response function (CCRF), $\{\Theta_m(t)\}_{m=1}^M$, which quantifies the possibility of each of the BCI choices to be the user's target character at time index, t . Data collection is continued until it is terminated at a stopping time, t_s , when the maximum CCRF value, $\Theta_{\max}(t)$, attains a threshold value, Θ_{th} , or the data collection limit, $t_{\max}$, is reached. The character that maximizes the CCRF at t_s is selected as the target character estimate, $\hat{C}^*$. The selected character, $\hat{C}^*$, can be used as feedback to inform the selection process for the next target character, e.g. via a language model.

Figure 5.

Distribution of TTI values for the (a) row-column, (b) checkerboard and (c) random stimulus presentation paradigms used in [47]. The means and standard deviations (error-bars) across participants ($n = 13$) are shown.

Figure 6.

Discriminability between the probability density functions (pdfs) of TTI-segregated scores and target classifier likelihood for row-column (RCP), random (RndP) and checkerboard (CBP) paradigms. Participant data from [47] was used for the analysis. The stimulus duration and ISI were both set to 62.5 ms for all paradigms. $D_{KL}(p||q)$, denotes the Kullback-Leibler divergence between pdfs p and q . p_i^j denotes the pdf a condition $i \in \{\text{TTI or } 1\}$, for TTI-segregated classifier scores and the aggregate target classifier likelihood, respectively, obtained from $j \in \{\text{train, test}\}$ set during cross-validation. Lower divergence indicates greater similarity. The means (solid lines) and standard deviations (shaded regions) across participants ($n = 13$) are shown. Results for TTI values with a low number of samples are not shown.

Figure 7.

Codebooks used in the online study, (a) row-column paradigm (RCP) and (b) performance-based paradigm (PBP), based on the interface shown in figure 2(a). Each column in the codebook represents a flash group, with characters presented highlighted in white. Each row represents the codeword for a character. Each column represents a flash group, with characters presented highlighted in white. The axis of randomization for a codebook is indicated in red: for the RCP, the order of presentation of the flash groups to maintain the character groupings of rows and columns; and for the PBP, the character-to-codeword assignments to maintain the TTI characteristics of the codebook. The example codebook for the RCP shown in (a) represents two instantiations of the base codebook shown in figure 2(b), with randomized flash group order.

Figure 8.

Online results with the Bayesian dynamic stopping algorithm for the row-column paradigm (RCP) and the performance-based paradigm (PBP): (a) mean character selection time, which includes a 3.5 second pause between character selections, (b) accuracy, (c) bit rate and (d) performance level, quantified by the detectability index, d . Participants were numbered in the order they were recruited. Results are ordered by increasing RCP accuracy. The mean participant results for each stimulus presentation paradigm are also displayed in the far right of each plot.

Figure 9.

Online and predicted performances with the Bayesian dynamic stopping algorithm for the PBP and RCP. (a) Accuracy, and (b) the expected stopping time, as a function of detectability index, d , with a given data collection limit, $t_{\max}$. Each scatter point represents a participant whose performance level is quantified by the detectability index. The line plots are the performance prediction curves obtained according to the method in [37]. The nested plot in (b) shows the EST for $d < 1$, in a log-space for visualization purposes.

Figure 10.

Kullback-Leibler divergence between a standard normal distribution, f_0 , and its horizontal transformations. The top panel shows f_0 and examples of its horizontal transformations, f_1 – f_4 . The bottom panel shows the KLD between q and f_0 , $D_{KL}(q||f_0)$, where q is a horizontal transformation of f_0 , with mean parameter μ_q . The line plot corresponds to the KLD as a function of μ_q . The scatter points correspond to the KLDs of f_1 – f_4 .

Table 1

Summary of participant online results.

Performance measure	RCP	PBP	<i>p</i> value [*]
Expected stopping time (flashes/character)	51.74 ± 16.50	39.59 ± 14.24	< 10 ⁻⁴
Mean character selection time (seconds)	9.97 ± 2.06	8.45 ± 1.78	< 10 ⁻⁴
Accuracy (%)	67.08 ± 22.51	74.96 ± 18.15	< 10 ⁻²
Bit rate (bits/min)	18.94 ± 12.90	24.93 ± 12.50	< 10 ⁻³
Detectability index	1.31 ± 0.55	1.37 ± 0.62	0.15

The mean (*M*) and standard deviation (*D*) for each performance measure are reported as $M \pm D$.

^{*} Statistical significance was tested using the Wilcoxon signed ranked test.