A Growing Bubble Speller Paradigm for Brain-Computer Interface Based on Event-related Potentials

Jing Jin*, *Senior Member, IEEE,* Xueqing Zhao, Ian Daly, Shurui Li, Xingyu Wang, Andrzej Cichocki, *Fellow, IEEE,* Tzyy-Ping Jung, *Fellow, IEEE*

***Abstract*— *Objective:* Event-related potentials (ERPs) reflect electropotential changes within specific cortical regions in response to specific events or stimuli during cognitive processes. The P300 speller is an important application of ERP-based brain-computer interfaces (BCIs), offering potential assistance to individuals with severe motor disabilities by decoding their electroencephalography (EEG) to communicate. *Methods:* This study introduced a novel speller paradigm using a dynamically growing bubble (GB) visualization as the stimulus, departing from the conventional flash stimulus (TF). Additionally, we proposed a "Lock a Target by Two Flashes" (LT2F) method to offer more versatile stimulus flash rules, complementing the row and column (RC) and single character (SC) modes. We applied the "Sub and Global" multi-window mode to EEGNet (mwEEGNet) to enhance classification and explored the performance of eight other representative algorithms. *Results:* Twenty healthy volunteers participated in the experiments. Our analysis revealed that our proposed pattern elicited more pronounced negative peaks in the parietal and occipital brain regions between 200 ms and 230 ms post-stimulus onset compared with the TF pattern. Compared to the TF pattern, the GB pattern yielded a 2.00% increase in online character accuracy (ACC) and a 5.39 bits/min improvement in information transfer rate (ITR) when using mwEEGNet. Furthermore, results demonstrated that mwEEGNet outperformed other methods in classification performance. *Conclusion and Significance:* These results underscore the significance of our work in advancing ERP-based BCIs.**

***Index Terms*—Brain-computer interface (BCI), event-related potential (ERP), speller paradigm, growing bubble, multiple windows.**

This work was supported by the Grant National Natural Science Foundation of China under Grant 62176090 and STI 2030-major projects 2022ZD0208900; in part by Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX. This research is also supported by Project of Jiangsu Province Science and Technology Plan Special Fund in 2022 (Key research and development plan industry foresight, fundamental research fund for the central universities JKH01231636 and key core technologies) under Grant BE2022064-1. (Corresponding author: Jing Jin).

Jing Jin, Xueqing Zhao, Shurui Li and Xingyu Wang are with the Key Laboratory of Smart Manufacturing in Energy Chemical Process, Ministry of Education, East China University of Science and Technology, Shanghai 200237, China (e-mail: jinjingat@gmail.com; xueqingzhao2021@163.com; shuruili1008@163.com; xywang@ecust.edu.cn).

Ian Daly is with the Brain-Computer Interfacing and Neural Engineering Laboratory, School of Computer Science and Electronic Engineering, University of Essex, CO4 3SQ Colchester, U.K. (e-mail: i.daly@essex.ac.uk).

Andrzej Cichocki is with the Systems Research Institute of Polish Academy of Sciences, 01-447 Warsaw, and Nicolaus Copernicus University (UMK), 87-100 Torun, Poland (e-mail: cichockiand@gmail.com).

Tzyy-Ping Jung is with the Swartz Center for Computational Neuroscience, Institute for Neural Computation, and the Center for Advanced Neurological Engineering, Institute of Engineering in Medicine, University of California San Diego (UCSD), La Jolla, CA 92093 USA. (e-mail: tpjung@ucsd.edu).

I. Introduction

BRAIN computer interfaces (BCIs) offer a direct communication pathway between the brain and the outside environment without depending on the human body's normal peripheral neural pathways and muscle tissue [1]. Electroencephalography (EEG), a non-invasive method widely used in BCI systems, acquires signals through electrodes placed on the scalp [2]. The acquired EEGs are translated into commands after pre-processing, feature extraction, and classification [3, 4]. The event-related potentials (ERPs) are cognitive potentials that result from the reception and processing of sensory information and the advanced processing of some cognitive activities, such as selective attention, memory updating, and semantic understanding [5]. ERPs include various potential components, such as N200 (N2), P300 (P3, including P3a and P3b), and N400 (N4).

Designing a good paradigm to evoke ERPs in data acquisition is imperative. As research on ERPs has advanced, the speller paradigm - which makes use of visual ERPs - has developed, and many studies have explored different stimulus elements and flash strategies. Regarding stimulus elements, Farwell et al. [6] first proposed a system using a 6×6 matrix stimulus interface comprising 26 letters and 10 digits, with rows and columns (RC) flashing in random order. Compared with the traditional 2-dimensional speller, Qu et al. [7] proposed a 3-dimensional P300 speller paradigm in which non-flashing characters had a blue background, and their background color turned green whenever the character flashed. Wu et al. [8] proposed a green circle and red dot spelling paradigm (GC-RD) to increase the amplitude of the P3a in the parietal area. Xu et al. [9] developed a speller based on miniature asymmetric visual evoked potentials (aVEPs), which encodes 32 characters with a space-code division multiple access scheme. It used very small and inconspicuous visual stimuli to implement an efficient BCI system. Kaufmann et al.

[10] used famous faces to cover characters transparently and found that these familiar faces evoked more ERP components and significantly reduced the sequence needed for correct character classification. As the study of the face paradigm gradually developed, the stimuli expanded to include facial emotions [11], cartoon and virtual faces [12], self-faces [13], and distinct color faces [14]. In addition, studies of motion-onset visual evoked potentials (mVEPs) based BCI spellers (also known as N200 spellers) have confirmed the validity of the non-flashing visual BCI paradigm [15, 16]. With flash strategies, Guan et al. [17] designed the single character (SC) paradigm to eliminate adjacent disturbances and double-flash problems. Jin et al. proposed a flashing pattern based on the binomial coefficient [18]. It considered the number of possible combinations given by the binomial coefficients to design the stimulus presentation, reducing the flashes per trial. The participants who used the lateral single-character speller (LSC) [19] needed to focus on only one side of the screen at a time. According to the study results, this left-right layout reduces local disturbances. The concepts of regions and multi-stages have also been added to the design to improve system performance [20-22].

Many studies have focused on ERP decoding and obtained significant results. Researchers widely use the linear discriminant analysis (LDA) classifier, which has been enhanced through a stepwise method called SWLDA [23], in ERP decoding. Introducing the shrinkage LDA (SKLDA) [24], which combined LDA with covariance shrinkage, has led to the detection of single-trial ERP signals and has shown superior classification performance compared to LDA and SWLDA classifiers. In addition, many studies have used the support vector machine (SVM) as a classifier [25, 26]. Li et al. [27] proposed an extreme gradient boosting based discriminant information mining (XGB-DIM) method for EEG classification. With the development of deep learning [28], convolutional neural networks (CNN), which have achieved state-of-the-art (SOTA) performance in computer vision (CV) [29, 30], are increasingly being applied to EEG decoding [31, 32]. Several studies have documented using different types of CNNs to detect ERPs. A robust and compact architecture proposed by Lawhern et al. [33], called EEGNet, used depthwise and separable convolutions to reduce network parameters and had neurophysiological interpretability. EEG-Inception [34] integrated inception modules to facilitate the extraction of feature maps at different temporal scales. Ma et al. [35] introduced the ERP-CapsNet framework, which used a capsule network (CapsNet) to extract the discriminative spatial-temporal ERP features and encode them in capsules to reduce the loss of valuable information. Wang et al. [36] combined capsule networks with temporal and spatial attention modules to propose ST-CapsNet. The phase preservation neural network (PPNN) employed a series of dilated temporal convolutional layers to capture temporal dynamics while preserving phase information [37].

Potentials evoked by the motion behavior of visual objects have the advantage of being insensitive to contrast and luminance compared to flash mode [15]. Manually determined flash rules currently have a limited variety. Enumeration with constraints provides a more comprehensive set of flash rules but demands more computational cost. For the ERP-based BCIs , spatial features of the EEG vary over time after stimulation onset. Setting different spatial feature weights for various periods is expected to enhance the decoding performance of the classifier. In this work, we explore the possibility of using a dynamically changing stimulus element in the speller and consider a more generalized approach to obtain flash rules. The classification step also introduces the “Sub and Global” multi-window mode. Here are the main contributions of this work: 1) we design a new growing bubble stimulus pattern using dynamically increasing radii instead of the typical flash; 2) we explore a more general flash strategy that uses two flashes to lock a unique target; 3) we use multi-window EEGNet (mwEEGNet) as the classifier for this system and present the classification results of eight other representative methods using data from both patterns. The experimental results showed that compared to the typical flash, our proposed pattern elicited more pronounced negative peaks in the parietal and occipital brain regions between 200 ms and 230 ms post-stimulus onset compared with the TF pattern and had better classification performance. Also, mwEEGNet outperformed other methods in classification.

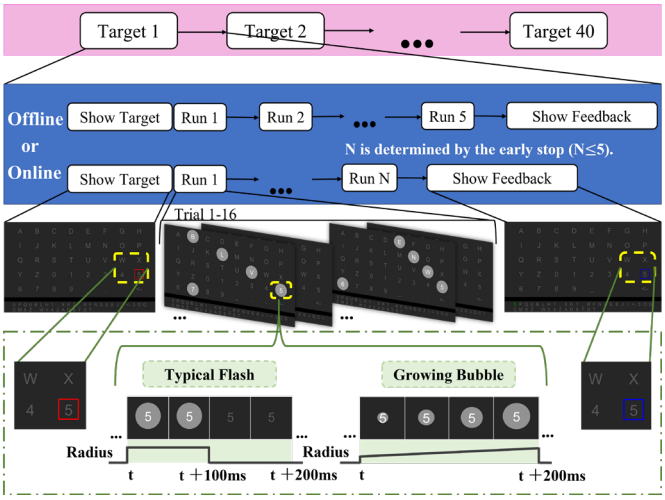

Fig. 1. Experimental procedure used with the TF and GB patterns.

II. Experiments And Methods

A. Experimental Paradigm

In our experiments, participants sat 60 cm in front of a 23-inch LED monitor (DELL P2314H) with a standard RGB color gamut, 1920×1080 pixels, and a 60 Hz refresh rate. The system ran on the Intel(R) Core(TM) i9-13900HX CPU, NVIDIA GeForce GTX 4060 with CUDA 11.8, and Python 3.8. Participants were asked to relax and avoid unnecessary movements during the experiment. The stimulation interface was constructed with Psychopy.

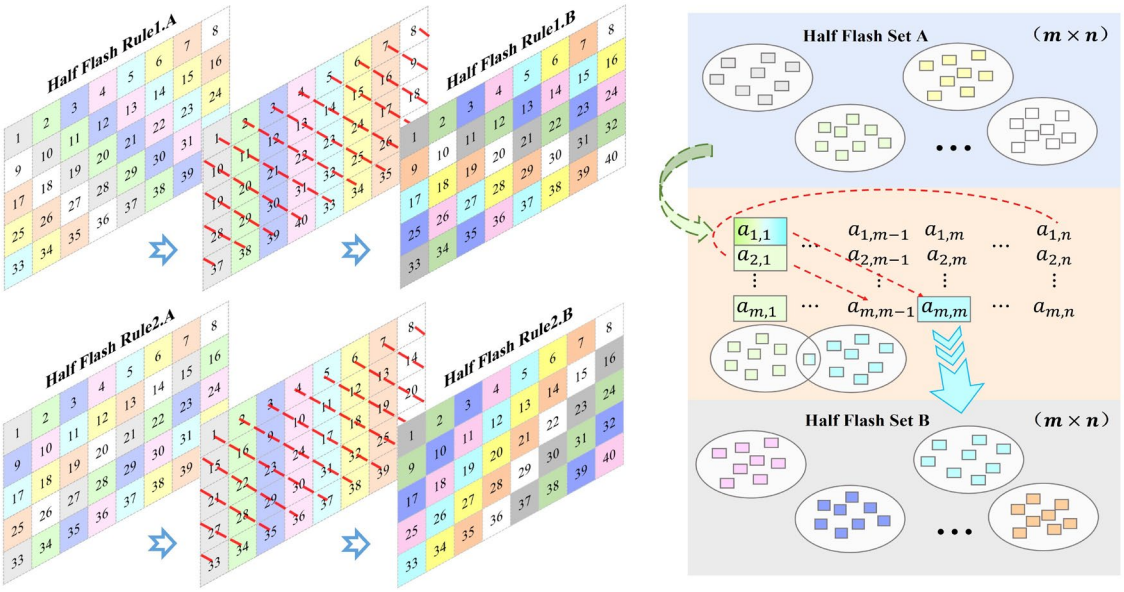

Fig. 2. The flowchart of LT2F to determine the flash rule.

TABLE I
STIMULATION INTERFACE PARAMETER SETTING

Name	Setting
Background	color = black
Rectangle	color = white, opacity = 0.15
Character 1	color = white, $\text{contrast}_{\text{unintensified}} = -0.3$, $\text{contrast}_{\text{intensified}} = 1$
Character 2	$\text{color}_{\text{default}}$ = white, $\text{contrast}_{\text{default}} = -0.3$, $\text{color}_{\text{matched}}$ = green, $\text{contrast}_{\text{matched}} = 1$, $\text{color}_{\text{mismatched}}$ = red, $\text{contrast}_{\text{mismatched}} = 1$
Circle (TF)	color = white, radius = h, opacity = 0.5
Bubble (GB)	color = white, radius $\in [0.4h, h]$, opacity = 0.5

Note: where Character 1 is the character in the spelling region, Character 2 is the character in the feedback, and h is the height of the characters.

This study contained two types of experiments: the typical flash (TF) speller and the growing bubble (GB) speller, as shown in Fig. 1. Each participant completed the two paradigm experiments on the same day, with a 15-minute rest between experiments. Both paradigms used a 5×8 grid to display stimuli but different intensification methods to highlight each stimulus. Each paradigm comprised both offline and online components, with participants taking a 10-minute break between these sections. Throughout the experiments, participants silently counted the number of flashes corresponding to the target character.

Each participant in the offline experiment had to spell 40 different targets. Each target comprised five runs, with each run consisting of 16 trials. An 800 ms interval separated each run. We set the stimulus onset asynchrony (SOA) to 200 ms. One intensification was regarded as a trial. At the onset of each new target, a red square indicated the target location for 2 seconds. After completing five runs, a 500 ms blue square provided feedback. The red and blue squares appeared at the target position during the offline phase. We used the offline data from each participant to train a personalized model, which was subsequently used for online decoding

Each participant was asked to spell 40 different character targets in the online experiment. For both patterns, the number of online runs was determined by the same early stopping strategy with the same parameters. Section II.E describes how to implement early stopping online. The stimulus interface provided character recognition results in the bottom area, with green displaying matching results and red displaying mismatches. However, to assess the effectiveness of various algorithms, we collected complete online data as the test set. Following an early stop, only the result was returned, but the flashes continued until five runs later when the blue square appeared at the recognition position.

Table I shows the stimulation parameters. In the GB pattern, the bubble (circle) radius linearly increases from $0.4h$ to h over 200 ms, where h is the height of the characters. The opacity of the bubble is 0.5, and its color is white. For the TF pattern, the circle's radius is h for the first 100 ms, and then decreases to 0 in the next 100 ms. When the characters are intensified, their color becomes white, and the contrast is 1. In other cases, their contrast is set to −0.3.

B. Flash Strategy

P300 BCI systems typically use a row/column (RC) [6] or single character (SC) [17] flash approach. Increasing the number of characters flashing simultaneously and reducing the number of flashes per run can enhance the information transfer rate (ITR), provided that paradigms maintain comparable identification accuracy. P300 spellers need at least two flashes to locate a character when multiple characters are intensified at once. In the 6×6 RC mode, a row and a column can lock only one specific character, ensuring precise character localization. Beyond the RC flashing format, a more generalized approach is required to ensure that two flashes can infer the position of a single character. Here, we introduce a method called Lock a Target by Two Flashes (LT2F).

Suppose the character matrix of the speller is $S \in \mathbb{R}^{s\times1}$, where $s = m\times n$ $(m \le n)$, m is the number of characters simultaneously intensified, and n represents half of the flash number. LT2F can be used to determine the rule with $2n$

flashes. These rules allocate different characters within the speller to distinct two-flash sets. The pseudocode and Fig. 2 illustrate the functioning of LT2F.

The elements within the same column of the rule matrix, Q, are intensified simultaneously. A run comprising $2n$ flashes corresponds to the rule matrix Q. Each Q contains two parts, $Q_{\text{half A}} \in \mathbb{R}^{m\times n}$ and $Q_{\text{half B}} \in \mathbb{R}^{m\times n}$. In the LT2F method, each character appears once in matrices $Q_{\text{half A}}$ and $Q_{\text{half B}}$ respectively. The half-flash rule $Q_{\text{half A}}$ can be either manually designated or randomly generated. The pseudocode's Steps 1 and 2 show the random generation approach. The elements in S are shuffled randomly and reshaped into the matrix $Q_{\text{half A}}$. It can also be further manually designed to avoid the adjacent disturbances. After $Q_{\text{half A}}$ is determined, the corresponding $Q_{\text{half B}}$ can be calculated via Steps 3-9 outlined in the pseudocode.

As shown in the right side of Fig. 2, following the diagonal direction determines the other half of the flash rule. This ensures that each flash in the second half-flash rule has at most one intersection with each flash in the first half-flash rule, and intersections cover all characters. The newly generated flash, comprising m elements, intersects exclusively with each of the m columns in $Q_{\text{half A}}$. A total of n generated flashes make up $Q_{\text{half B}}$.

The left side in Fig. 2 shows a part of the flash rule used in this study where $m=5$, $n=8$, and $Q \in \mathbb{R}^{5\times 16}$. The numbers 1-40 within the matrix denote element indices, while blocks of the same color in the Half Flash Rule denote elements within the same flash. We generate different Q using LT2F and randomly shuffled $Q_{\text{half A}}$ and $Q_{\text{half B}}$ in each case. To avoid the target appearing in the adjacent flash, the Q whose last column of $Q_{\text{half A}}$ has an empty intersection with the first column of $Q_{\text{half B}}$ is considered a satisfied flash rule and put in a flash rule pool comprising 500 entries. Subsequently, for each run, the flash rule matrix Q is randomly from the flash rule pool.

Algorithm 1 Lock a Target by Two Flashes (LT2F)

Input: Element (character) vector $S \in \mathbb{R}^{s\times 1}$, number of elements flashing simultaneously m, number of flashes in a trial $2n$.

Output: Flash rule matrix $Q \in \mathbb{R}^{m\times 2n}$, where Q consists of two parts, $Q_{\text{half A}} \in \mathbb{R}^{m\times n}$ and $Q_{\text{half B}} \in \mathbb{R}^{m\times n}$.

1: $S \leftarrow$ random shuffle S
2: $Q_{\text{half A}} \leftarrow$ reshape S into $[m, n]$
3: **for** $i_n \leftarrow 1$ **to** n **do**
4: $\quad tmp \leftarrow$ zeros $([m,1])$
5: $\quad$ **for** $i_m \leftarrow 1$ **to** m **do**
6: $\quad\quad tmp[i_m] \leftarrow Q_{\text{half A}}[i_m,\ (i_m-1)\%n+i_n]$
7: $\quad$ **end**
8: $\quad Q_{\text{half B}}[:,\ i_n] \leftarrow tmp$
9: **end**
10: $Q \leftarrow$ concatenate $Q_{\text{half A}}$ and $Q_{\text{half B}}$ in the 2nd dimension
11: **return** Q

C. Participants and Data Acquisition

We recruited twenty healthy volunteers (P1-P20, 21-28 years old, 6 females, and 14 males) with normal or corrected to normal vision to participate in the study. All participants are right-handed and have normal color vision. Their native language is Mandarin, and they are familiar with the characters on the display. The Local Institutional Review Board approved the experimental procedures (Document Number: ECUST-2022-054). Each participant provided written informed consent after thoroughly explaining the study's objectives, tasks, and potential consequences of participation. Among the participants, nine (P6, P8, P9, P13, P15, P16, P17, P18, and P20) prior experience with BCI experiments.

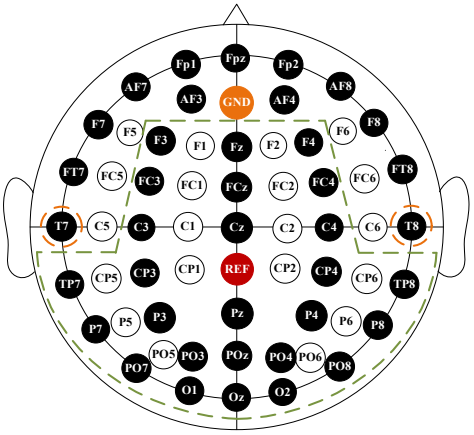

Fig. 3. Channel locations of NSW364.

The experiments used a wireless EEG acquisition system NSW364 (Neuracle, NeuSen W series, 59 EEG, 4EOG, 1ECG) to acquire data at the 1000 Hz sampling rate with Ag/AgCl electrodes (channels) placed at the positions following the 10-20 system, as shown in Fig. 3. We maintained electrode impedance below 20 kΩ. The reference channel is CPz and the ground channel is AFz. Of the 39 black channels shown in Fig. 3 for data acquisition, 26 were used for EEG decoding, based on existing literature [14, 24]. We used channels T7 and T8 for re-reference.

D. Decoding Process

In this study, the decoding workflow includes pre-processing and classification algorithms. Pre-processing comprised re-referencing the stored EEG signals using the average data from T7 and T8 and down-sampling them to 250 Hz. We used an IIR notch filter centered at 50 Hz to remove power line interference. A 4th-order Butterworth filter was used for band-pass filtering between 1 and 40 Hz to reduce the noise in the EEG. We extracted data segments from the 0-1 s after stimulus onset for the classification of each trial, and we corrected the baseline using data from the 100 ms preceding stimulus. The EEG signals were then standardized using z-scoring.

Here, we use the multi-window EEGNet as the classifier, called mwEEGNet. This model recognizes the targets from spellers by classifying the EEG features after pre-processing. Fig. 4 and Table II illustrate the detailed model structure.

The input features undergo decomposition into multiple overlapping sub-windows using a sliding method in the first

TABLE II
THE ILLUSTRATION OF THE MWEEGNET ARCHITECTURE

Block	Type	Filter	Kernel	Output	Activation	Options
	Input			$(1,\ C,\ T_i)$		
1	Conv2D	F_1	$(1,\ T_i//2)$	$(F_1,\ C,\ T_i)$	Linear	padding = same
	BatchNorm			$(F_1,\ C,\ T_i)$		
2	DepthwiseConv2D	$D*F_1$	$(C,\ 1)$	$(D*F_1,\ 1,\ T_i)$	Linear	depth = D, max norm = 1 padding = valid
	BatchNorm			$(D*F_1,\ 1,\ T_i)$		
	Activation			$(D*F_1,\ 1,\ T_i)$	ELU	
	AveragePool2D		$(1,\ B_1)$	$(D*F_1,\ 1,\ T_i//B_1)$		$B_1 = 2$ (sub) or 4 (global)
	Dropout			$(D*F_1,\ 1,\ T_i//B_1)$		$p = 0.5$
3	SeparableConv2D	$D*F_1$	$(1,\ T_i//B_1//K)$	$(D*F_1,\ 1,\ T_i//B_1)$	Linear	$K = 1$ (sub) or 2 (global) padding = same
	BatchNorm			$(D*F_1,\ 1,\ T_i//B_1)$		
	Activation			$(D*F_1,\ 1,\ T_i//B_1)$	ELU	
	AveragePool2D		$(1,\ B_2)$	$(D*F_1,\ 1,\ T_i//B_1//B_2)$		$B_2 = 4$ (sub) or 8 (global)
	Dropout			$(D*F_1,\ 1,\ T_i//B_1//B_2)$		$p = 0.5$
Out	Flatten			$(D*F_1*\sum_{i=1}^{5}(T_i//B_1//B_2))$		
	Dense			N_{class}	Softmax	max norm = 0.25

Note: where C = number of channels, T_i = number of time points in the ith window ($i \in [1, 4]$ for sub-windows, $i = 5$ for global window), F_1 = number of temporal filters, D = depth multiplier (number of spatial filters), B_1 = width of the average pool layer in block 2 (2 for sub-windows, 4 for the global window), B_2 = width of the average pool layer in block 3 (4 for sub-windows, 8 for the global window), K is the parameter that decides the size of the kernel, and N_{class} = number of classes.

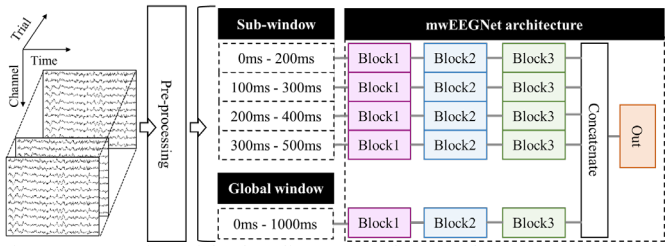

Fig. 4. An overview of the decoding process framework.

step, starting from 0 ms to 500 ms with a width of 200 ms and an overlap of 100 ms. Consequentially, the mwEEGNet classifier considers a total of 4 sub-windows: 0-200 ms, 100-300 ms, 200-400 ms, and 300-500 ms. All windows with C channels and T_i time samples are imported into the network framework in parallel. The model uses the Adam optimizer [38] and minimizes the binary cross-entropy loss function. A manual rescaling weight is given to the loss of each batch element to mitigate the imbalance of sample categories. In this study, the weight of loss is [0.125, 0.875] and the learning rate is 0.001. We run 200 training iterations with a batch size of 512 using Pytorch [39]. As shown in Table II, the mwEEGNet architecture comprises four blocks. The data from each window undergo sequential processing in Blocks 1-3 blocks and are then concatenated before entering the Out block.

Block 1 comprises a temporal 2D convolution layer (Conv2D) and a batch normalization layer (BN) [40]. The kernel size $(1,\ T_i//2)$ is chosen to be half of the input window length T_i, where $i \in [1, 5]$ is the number of windows (4 sub-windows and a global window) and F_1 is 8, indicating the number of temporal filters.

Block 2 works as a spatial filter and has a depthwise convolution (DepthwiseConv2D) [41] to weigh the selected channels with kernel size $(C,\ 1)$ (where $C = 26$ is the number of used channels when decoding) and the depth parameter $D = 2$. Feature maps after DepthwiseConv2D are sent to BN, then through the exponential linear unit (ELU) [42] activation function. An average pooling layer is used for temporal dimension reduction and B_1 is set to 2 for sub-windows or 4 for the global window. The dropout [43] probability is set to 0.5 to prevent overfitting when training on small sample sizes.

A separable convolution [41] consisting of DepthwiseConv2D of size $(1,\ T_i//B_1//K)$ and point convolution is included in Block 3, where K is a hyperparameter that controls the convolution kernel size. For the global window, we still consider setting the kernel size to half of the input feature map of this layer in the time dimension $(K = 2)$ and using the average pooling layer of size (1, 8) to reduce the dimensionality. For sub-windows, the kernel width is equal to the feature map width of the input to this layer $(K = 1)$, and the size of the average pooling layer is (1, 4).

As shown in Fig. 4, Block Out performs the classification process after concatenating the output features of Block 3.

E. Online Adaptive Setting

We set up an adaptive strategy in the online system to achieve the early stop. As shown in Fig. 1, all character probabilities are averaged over multiple runs for each target. When the maximum character probability is not less than 1.5

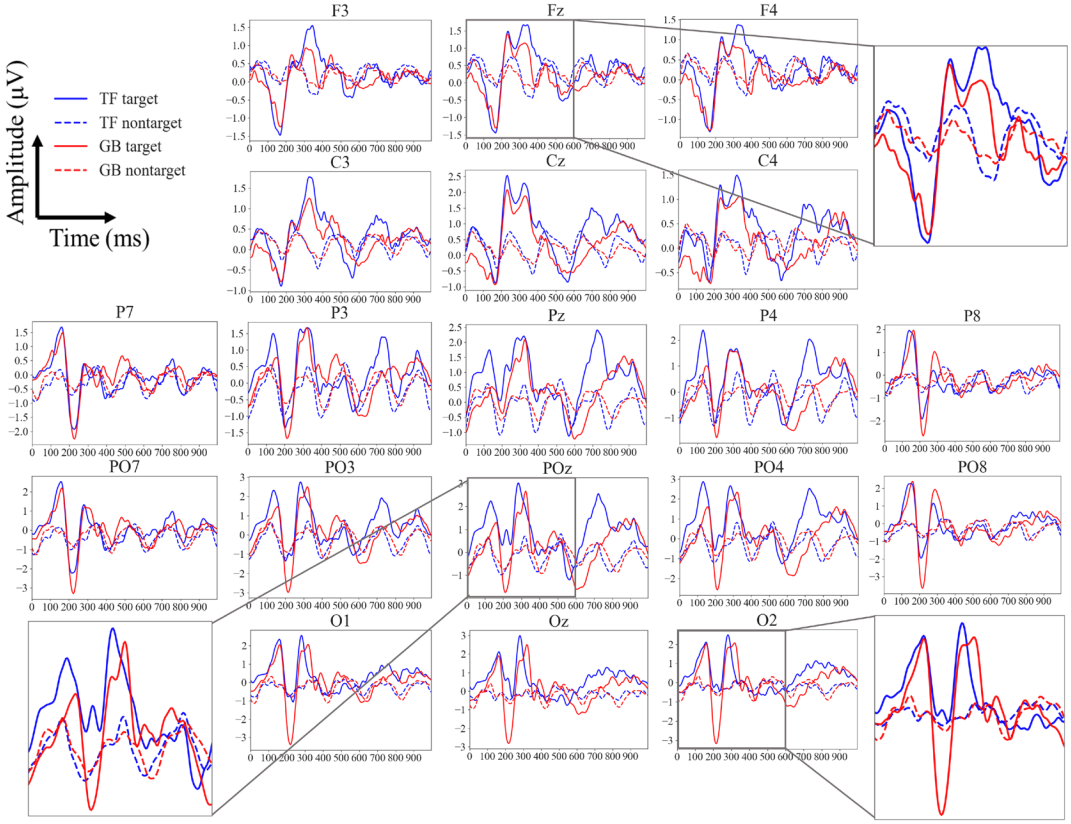

Fig. 5. The grand-averaged ERP amplitudes of 20 participants over 19 channels in two spelling paradigms.

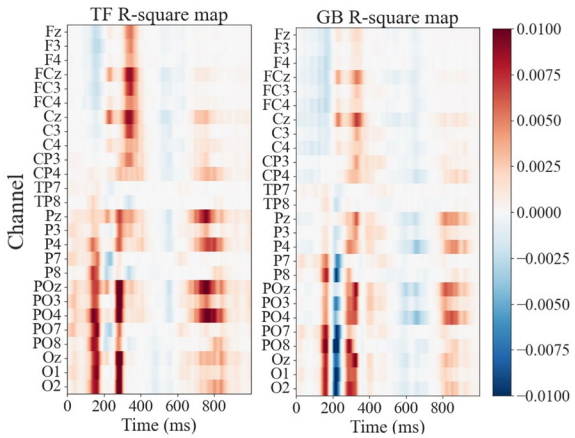

Fig. 6. The signed R-squared value maps from 0 s to 1 s of 20 participants over 26 channels in the TF and GB patterns.

times the second largest character probability, we consider the recognition reliable, and a new run for the same target is unnecessary. The maximum number of runs is set to 5, which is the same as the offline number of runs. If the confidence criterion is not met after 5 runs, the algorithm will directly output the character corresponding to the maximum average probability.

F. Evaluation and Statistical Analysis

The classification accuracy (ACC), F1-score (F1), average run number (AVR), and information transfer rate (ITR) [44] are calculated to measure the performance of the TF and the GB pattern. The ITR is defined as:

$$\mathrm{ITR}=\left(\log_2 M+P\log_2 P+(1-P)\log_2\frac{1-P}{M-1}\right)\frac{60}{T} \qquad (1)$$

where M is the number of character classes (here, $M=40$), P represents the character classification accuracy, and $T=2.5+4j$ $(1\le j\le 5)$ is the time required to obtain a character. There is a 2.5 s interval between each target character, including 500 ms for feedback presentation and 2s for the next target cue. Each trial lasts 200 ms, and a single run comprising 16 trials takes 4 s.

For statistical analysis of the experimental outcomes, we used IBM SPSS Statistics, version 25. The Shapiro-Wilk (S-W) test is used to check whether the sample conforms to a normal distribution. Given that most indicators did not exhibit a normal distribution, we chose the Wilcoxon signed-rank test to statistically analyze the system's performance.

III. Experimental Results

A. ERP Analysis

We visualized the offline data from 20 participants after pre-processing without applying z-scoring. Selected channels in the frontal, central, parietal, and occipital regions were included for display. Fig. 5 shows the grand-averaged amplitudes of target and nontarget trials for each of the two stimulus patterns. Bimodal waves were observed on channels F3, Fz, F4, C3, Cz, and C4 between 210 ms and 350 ms in both patterns. The P100 component predominantly manifested in the parietal and occipital regions, while the P300 distribution

extended along the zero-line channels, with amplitudes exceeding 2 μV on channels Cz, Pz, POz, PO3, PO4, Oz, O1, and O2.

The P300 components of the GB pattern in the frontal and central region had lower amplitudes than those evoked by the adjsmaller than that of the TF pattern. The GB pattern showed stronger negative deflections in the parietal and occipital regions from 200 ms to 250 ms than the TF pattern, especially in channels Oz, O1, and O2. Based on existing studies [15, 16, 45], we conclude that small to large stimulus patterns of GB create motion-specific stimuli that lead to this phenomenon.

Fig. 6 shows the signed R-squared values of ERPs from 0 s to 1 s averaged over all the participants. The formula is given as follows:

$$r = \frac{\sqrt{N_{\text{target}} N_{\text{nontarget}}}}{N_{\text{target}} + N_{\text{nontarget}}} \cdot \frac{\text{mean}(X_{\text{target}}) - \text{mean}(X_{\text{nontarget}})}{\text{std}\left(X_{\text{target}} \cup X_{\text{nontarget}}\right)} \quad (2)$$

where N_{target} and $N_{\text{nontarget}}$ refer to the number of targets and nontargets, respectively, and X_{target} and $X_{\text{nontarget}}$ are the features of the corresponding classes. Then, $R^2 = \text{sgn}(r) r^2$. The signed R-squared maps also indicated that greater discriminative negative components can be evoked by GB at about 200-230 ms.

B. *Offline Analysis*

We evaluated the performance using mwEEGNet through 5-fold cross-validation, considering both single-trial and character offline classification results.

Fig. 7 (a) presents each participant's single-trial average ACC and F1 scores after cross-validation. Most participants obtained higher ACC and F1 scores with the GB pattern. On average, participants attained ACC and F1 scores of 97.12% and 93.08%, respectively, with the GB pattern, compared to 96.14% and 90.69% with the TF pattern. Fig. 7 (b) and (c) depict the average offline character ACC and ITR scores as the number of runs increases. The GB pattern showed higher ACC and ITR scores with the same number of runs. When decoding with only the first run of each character, the GB pattern achieved an average character ACC of 88.35% and an ITR of 39.96 bits/min, while the TF pattern achieved an average character ACC of 86.00% and an ITR of 38.27 bits/min. Notably, the GB pattern achieved a character accuracy of 99.30% using 3 runs of data.

C. *Online Analysis*

Each participant attended an online spelling task of 40 characters for both patterns after offline training. The online system used an early stopping strategy to reduce the number of runs of a single character. Table III shows the online single-trial and character recognition results. The calculated *p*-values indicate the significance of the differences between each evaluated metric pair.

When using mwEEGNet as a classifier, the single trial ACC

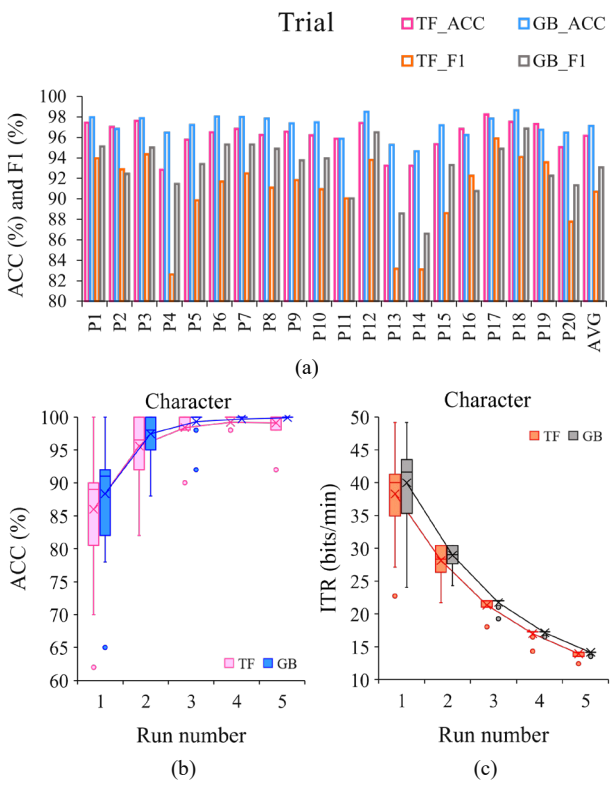

Fig. 7. The offline performance of the TF and GB patterns including (a) the ACC (%) and F1 (%) scores based on trial classification, (b) the ACC (%) of character recognition and (c) the ITR (bits/min).

in the GB pattern was significantly higher than that achieved by the TF pattern, with a difference of 1.23%. The online ITR of the GB pattern achieved 32.49 bits/min on average, with a peak value of 39.42 bits/min, and the average run number used by participants was 1.87. The participant, P14, who performed poorly during offline cross-validation, had a lower ITR of 19.69 bits/min for the online experiment.

IV. Discussion

In this study, we developed the GB speller paradigm using mwEEGNet as the classifier. This paradigm used the LT2F method to arrange the stimulus order within each run. The relevance of the input features to the predicted results was visualized to provide interpretability. The rationale for incorporating multi-windows stems from observing temporal changes in the cortical regions activated by target stimuli, potentially resulting in varying channel weights. We examined the weight distribution across each window and assessed how the number of windows impacted the mwEEGNet classifier's performance. Subsequently, we delved into comparing the performance of nine distinct classifiers across the two paradigms to underscore the system and algorithm effectiveness. The offline data served as the training set, while all online data constituted the test set. Finally, we analyzed the limitations of this study.

TABLE III
ONLINE ACC, F1, AVR, ITR, AND SIGNIFICANCE COMPARISON FOR THE TF AND GB PATTERNS FOR ALL PARTICIPANTS

Participant	Trial [TF, GB]		Character [TF, GB]			
	ACC (%)	F1 (%)	ACC (%)	F1 (%)	AVR	ITR (bits/min)
P1	[96.91, **97.29**]	[92.56, **93.46**]	[100.00, 100.00]	[100.00, 100.00]	[2.13, **1.68**]	[29.03, **34.71**]
P2	[96.21, **97.04**]	[90.73, **92.95**]	[97.50, 97.50]	[96.67, 96.67]	[2.10, **1.90**]	[27.64, **29.83**]
P3	[97.09, **97.86**]	[93.19, **95.13**]	[100.00, 100.00]	[100.00, 100.00]	[1.83, **1.75**]	[32.58, **33.61**]
P4	[93.48, **95.88**]	[84.04, **90.12**]	[97.50, **100.00**]	[96.67, **100.00**]	[2.93, **2.05**]	[21.22, **29.84**]
P5	[96.02, **96.79**]	[90.83, **92.10**]	[97.50, **100.00**]	[96.67, **100.00**]	[**1.93**, 1.95]	[29.54, **31.00**]
P6	[97.40, **98.88**]	[93.83, **97.41**]	[100.00, 100.00]	[100.00, 100.00]	[1.80, **1.40**]	[32.92, **39.42**]
P7	[96.52, **97.37**]	[92.21, **93.66**]	[100.00, 100.00]	[100.00, 100.00]	[1.98, **1.73**]	[30.70, **33.97**]
P8	[96.41, **96.83**]	[91.71, **92.26**]	[100.00, 100.00]	[100.00, 100.00]	[2.18, **1.93**]	[28.51, **31.31**]
P9	[96.23, **98.12**]	[90.86, **95.52**]	[100.00, 100.00]	[100.00, 100.00]	[1.95, **1.58**]	[31.00, **36.29**]
P10	[94.74, **96.33**]	[87.80, **90.94**]	[97.50, **100.00**]	[96.67, **100.00**]	[2.53, **1.88**]	[23.91, **31.93**]
P11	[95.30, **95.56**]	[88.96, **89.42**]	[95.00, **97.50**]	[93.33, **96.67**]	[2.63, **2.33**]	[22.02, **25.53**]
P12	[96.78, **98.44**]	[92.12, **96.32**]	[97.50, **100.00**]	[96.67, **100.00**]	[1.70, **1.40**]	[32.39, **39.42**]
P13	[88.79, **97.21**]	[70.96, **93.34**]	[75.00, **100.00**]	[70.00, **100.00**]	[3.63, **1.85**]	[11.26, **32.25**]
P14	[**93.62**, 93.41]	[**84.24**, 83.18]	[97.50, 97.50]	[96.67, 96.67]	[3.63, **3.20**]	[17.72, **19.69**]
P15	[94.58, **96.60**]	[87.31, **91.89**]	[100.00, 100.00]	[100.00, 100.00]	[3.03, **2.03**]	[21.87, **30.12**]
P16	[**95.81**, 95.12]	[**90.26,** 88.40]	[100.00, 100.00]	[100.00, 100.00]	[**2.20**, 2.40]	[**28.26**, 26.39]
P17	[**97.89**, 97.85]	[**95.17**, 95.03]	[100.00, 100.00]	[100.00, 100.00]	[1.78, **1.53**]	[33.26, **37.13**]
P18	[97.83, **98.17**]	[94.84, **95.67**]	[100.00, 100.00]	[100.00, 100.00]	[1.80, **1.45**]	[32.92, **38.47**]
P19	[96.73, **97.32**]	[92.07, **93.64**]	[97.50, **100.00**]	[96.67, **100.00**]	[2.10, **1.58**]	[27.64, **36.29**]
P20	[95.81, **96.75**]	[89.97, **92.38**]	[100.00, 100.00]	[100.00, 100.00]	[2.28, **1.83**]	[27.53, **32.58**]
AVG	[95.71, **96.94**]	[89.68, **92.64**]	[97.63, **99.63**]	[97.00, **99.50**]	[2.30, **1.87**]	[27.10, **32.49**]
STD	[2.04, 1.27]	[5.34, 3.22]	[5.53, 0.92]	[6.66, 1.22]	[0.58, 0.42]	[5.82, 4.94]
p	<0.001	0.001	0.011	0.014	<0.001	<0.001

Note: AVG refers to the average value, and STD refers to the standard deviation. p-values are calculated by the Wilcoxon signed-rank test, and samples are significantly different from each other when $p \leq 0.05$. For different metrics, the better-performing data are indicated in bold.

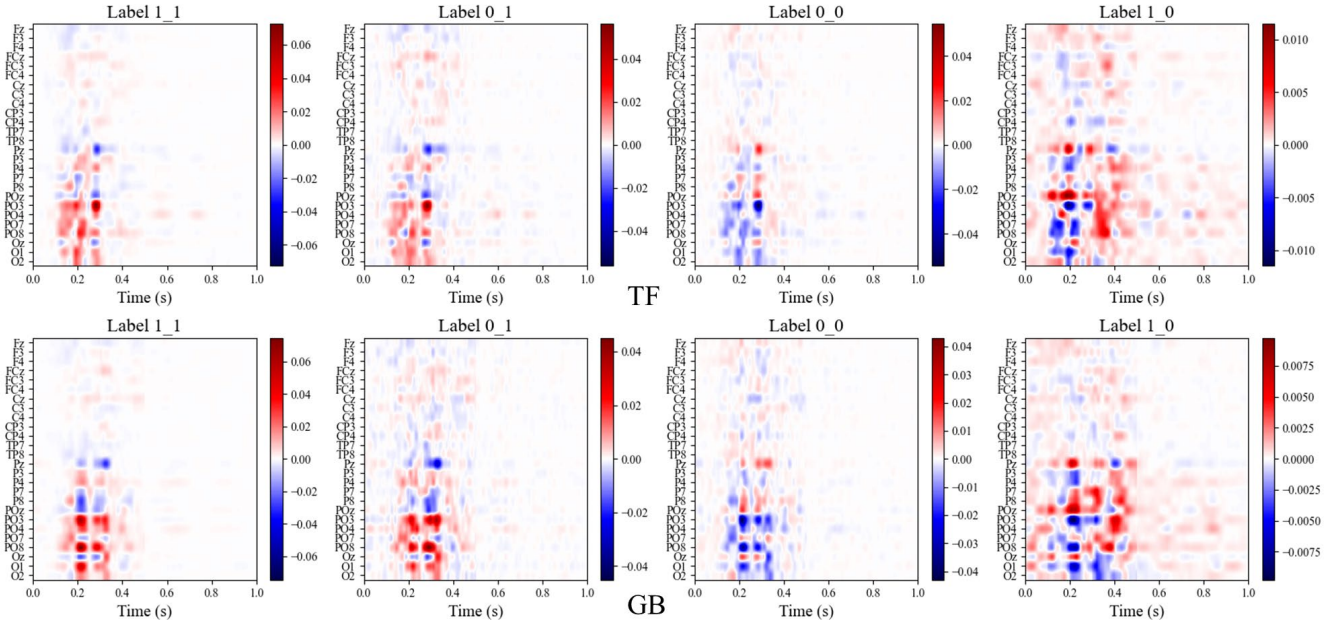

Fig. 8. The participant-averaged feature relevance for the trained mwEEGNet in the TF and GB patterns by using DeepLIFT. Label *i_j* indicates that the true label is Class *i* and the predicted label is Class *j*. Class 1 represents the target, and Class 0 is the nontarget.

A. Feature relevance on the classification decision when using mwEEGNet

We used DeepLIFT [46] to demonstrate the importance of each input feature to the model output. Positive relevance values indicate evidence supporting the prediction, while negative relevance values indicate evidence against the prediction [33]. As shown in Fig. 8, TF and GB obtained stronger relevance in the parietal and occipital regions at about 100-400 ms when features were correctly predicted (Label 1_1

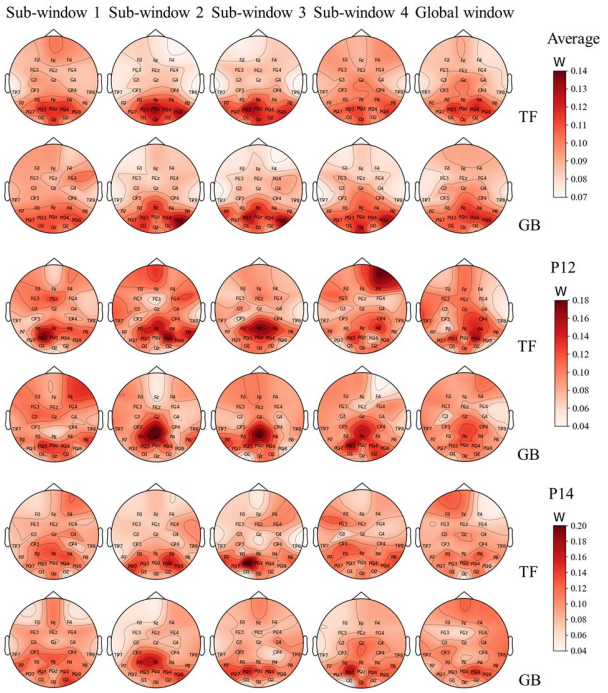

Fig. 9. Topographies of spatial convolution weights on different windows in the DepthwiseConv2D layer. The Average weights are calculated by taking the average of the absolute values from all the participants' filters. The weights for P12 and P14 are calculated by averaging the absolute values of self filters' weights.

and 0_0). When incorrectly predicted (Label 1_0 and Label 0_1), the feature relevance values were more widely distributed on the scalp. Around 200-230 ms, Label 1_1 of GB exhibited stronger feature relevance. This aligns with the ERP analysis in Section III.A.

B. The Weights of the Spatial Convolution in Different Window

We used MNE [47] to visualize the weights $W \in \mathbb{R}^{C\times 1}$ of the DepthwiseConv2D layer within the mwEEGNet classifier. Fig. 9 shows the spatial weights of the participants' averages, with P12 exhibiting higher ITR and P14 displaying lower ITR. The Average weights were calculated by taking the average of the absolute values across all the participants' filters. The weights for P12 and P14 were calculated by averaging the absolute values of the respective self-filters' weights. We segmented the data into five time windows: 0-200 ms, 100-300 ms, 200-400 ms, 300-500 ms, and 0-1 s. Across both paradigms, the weights within each window for the Average group showed elevated values in the parietal and occipital regions compared to other cortical regions. Spatial patterns fluctuate across time windows. For P12, Pz exhibited notably higher weights in sub-windows 2 and 3, with this trend less prominent in other windows. Conversely, in P14, PO3 and PO4 exhibited higher weights in sub-windows 2 and 3 within the TF pattern, while sub-window 2 in the GB pattern emphasized P3 and Pz more.

C. The Effect of Multiple Windows on mwEEGNet

We used the offline data as the training set and the whole online data as a test set to compare the classification performance with different numbers of windows. The "Sub and Global" mode was used to design mwEEGNet. We segmented the entire data into nine sub-windo ws by extracting data from 0 to 1 s post-stimulus onset, with a window length of 200 ms and a stride of 100 ms. As the number of sub-windows N_{SW} increased, the network parameters grew linearly. Table IV shows the effect of different sub-window numbers on the performance of mwEEGNet, where $N_{SW}=1$ means the inclusion of the 0-200 ms window, $N_{SW}=2$ means the inclusion of the windows 0-200 ms and 100-300 ms. The network parameters increased by 2232 for each additional window. As shown in Table IV and Table V, the used method achieved higher trial ACC and F1 scores than those achieved by EEGNet ($N_{SW}=0$). However, with further window additions ($N_{SW}\geq 4$), the enhancement in classification performance plateaued and, in some cases, experienced a slight decline, albeit still surpassing EEGNet. In this study, we used 4 sub-windows based on the preliminary testing of paradigms. According to our study, the early visual component and the P300 component, which are key features of the two paradigms, predominantly manifest during the first 0-500ms of visual perception and conclude before 600 ms.

D. Comparison of Classification Performance with Other Classifiers

We assessed the performance of various classifiers across two paradigms, encompassing three traditional machine learning methods and six neural networks. For XGBDIM, we used the multi-XGBDIM function from the open-source code. The channel combination was set as follows: [Fz, F3, F4, FCz, FC3, FC4, Cz, C3, C4], [TP7, CP3, Pz, P7, P3, POz, PO7, PO3, Oz], [TP8, CP4, Pz, P8, P4, POz, PO8, PO4, Oz], and [P3, Pz, P4, PO3, POz, PO4, O1, Oz, O2]. All other parameters were kept at default. To address data imbalance $N_{\text{nontarget}}:N_{\text{target}}=7{:}1$, we set the weight of the cross-entropy to [0.125, 0.875]. The training involved 200 iterations (epochs) for these models, with a batch size of 512 for EEGInception, PPNN, and EEGNet. ERPCapsNet and STCapsNet, however, used batch sizes of 128 and 64, respectively, based on findings from the original paper indicating inferior classification results with a batch size of 512. The Adam optimizer was used for training, with a learning rate of 0.001.

As shown in Table V, classifiers generally obtained better classification performance in the GB pattern than in the TF pattern. The results of significance tests showed that mwEEGNet achieved statistically significant improvements relative to SKLDA, SVM, XGBDIM, ERPCapsNet, STCapsNet, PPNN, and EEGNet in trial ACC, trial F1, and character ACC. Furthermore, mwEEGNet demonstrated significant superiority over EEGInception in both trial ACC and F1 scores. Fig. 10 shows the ITRs of different algorithms for the TF and GB patterns, with mwEEGNet exhibiting the best performance. Table VI shows the computational

TABLE IV
PARTICIPANT-AVERAGED TRIAL ACC (%), TRIAL F1 (%), CHARACTER ACC (%), AND ITR (BITS/MIN) FOR THE TF AND GB PATTERNS WHEN USING DIFFERENT NUMBERS OF SUB-WINDOWS

Number of sub-windows N_{sw}		1	2	3	4	5	6	7	8	9
Params		4706	6938	9170	11402	13634	15866	18098	20330	22562
TF	Trial ACC	95.17	95.75	95.73	**95.83**	95.82	95.72	95.68	95.70	95.67
	Trial F1	87.85	89.55	89.61	89.97	**89.98**	89.68	89.69	89.68	89.65
	Char ACC 1	78.75	83.13	**83.38**	83.25	82.25	83.00	81.50	82.50	83.25
	Char ACC 2	91.25	92.75	93.25	92.88	**93.88**	93.00	93.00	92.38	92.38
	Char ACC 3	95.50	96.38	95.88	95.75	**96.63**	96.00	95.63	96.00	95.13
	Char ACC 4	96.63	97.88	97.63	98.00	97.50	**98.13**	97.25	97.75	97.00
	Char ACC 5	97.38	98.25	98.00	98.25	97.75	98.00	98.13	**98.50**	98.13
	ITR 1	32.51	35.62	**35.79**	35.57	34.84	35.57	34.45	35.05	35.71
	ITR 2	25.90	26.78	26.95	26.65	**27.30**	26.90	26.90	26.51	26.53
	ITR 3	20.31	20.71	20.51	20.39	**20.72**	20.58	20.41	20.55	20.24
	ITR 4	16.22	16.62	16.56	16.63	16.51	**16.76**	16.49	16.59	16.36
	ITR 5	13.55	13.76	13.71	13.76	13.65	13.74	13.74	**13.83**	13.76
GB	Trial ACC	96.35	96.58	96.87	97.01	97.00	97.00	**97.03**	96.85	96.84
	Trial F1	90.93	91.68	92.39	92.82	92.85	92.83	**92.96**	92.49	92.50
	Char ACC 1	86.38	86.00	87.13	88.88	89.38	**90.00**	88.88	87.50	88.00
	Char ACC 2	96.75	96.00	97.63	**98.25**	97.75	97.50	97.25	97.50	97.75
	Char ACC 3	98.38	98.88	**99.25**	98.75	99.13	98.88	98.88	99.00	99.00
	Char ACC 4	99.13	99.38	99.50	**99.75**	99.50	**99.75**	99.50	99.63	99.63
	Char ACC 5	99.50	99.38	**99.75**	**99.75**	99.63	**99.75**	99.50	99.50	99.63
	ITR 1	37.61	37.30	38.17	39.43	39.80	**40.45**	39.52	38.49	38.94
	ITR 2	28.46	28.08	28.98	**29.37**	29.05	28.94	28.79	28.95	29.00
	ITR 3	21.27	21.51	**21.67**	21.46	21.61	21.50	21.49	21.55	21.56
	ITR 4	16.95	17.02	17.08	**17.16**	17.08	**17.16**	17.07	17.12	17.11
	ITR 5	14.04	14.00	**14.11**	**14.11**	14.07	**14.11**	14.03	14.04	14.07

Note: where $N_{sw}=1$ means including the 0-200 ms window, $N_{sw}=2$ means including the 0-200 ms and 100-300 ms windows, and so on. Char ACC *i* and ITR *i* (*i* = 1, 2, 3, 4, 5) represent the character ACC and ITR obtained through the first *i* runs. Params represents the number of trainable parameters in the model when assuming an input size of (512, 1, 26, 250). The maximum of each row is indicated in bold.

TABLE V
PARTICIPANT-AVERAGED TRIAL ACC (%), TRIAL F1 (%), AND CHARACTER ACC (%) ACHIEVED BY THE DIFFERENT CLASSIFIERS FOR THE TF AND GB PATTERNS

		SKLDA	SVM	XGBDIM	ERPCapsNet	STCapsNet	EEGInception	PPNN	EEGNet	mwEEGNet
Trial ACC	TF	93.99***	91.98***	87.42***	93.94***	93.73***	94.99***	94.74***	95.07***	**95.83**
	GB	95.32***	93.28***	89.77***	95.34***	95.02***	96.58**	96.41***	96.35***	**97.01**
Trial F1	TF	85.64***	81.50***	77.75***	84.67***	84.46***	88.67**	86.67***	87.41***	**89.97**
	GB	88.64***	84.31***	80.94***	88.34***	87.63***	92.16*	91.24***	90.80***	**92.82**
Char ACC 1	TF	73.88**	63.50***	67.00***	68.88***	70.50***	81.50	77.38**	78.25**	**83.25**
	GB	78.75***	70.63***	71.13***	77.00***	76.38***	86.88	85.38*	86.00*	**88.88**
Char ACC 2	TF	87.88**	79.38***	84.25***	84.50**	86.25**	90.75	89.25**	91.00	**92.88**
	GB	92.00***	86.50***	87.50***	92.75***	91.75***	96.50**	96.88*	95.75**	**98.25**
Char ACC 3	TF	92.38**	86.63***	89.25**	91.00**	90.75**	94.75	93.38*	95.25	**95.75**
	GB	94.88**	92.63**	93.13**	96.75**	95.50**	98.75	97.88*	98.13	**98.75**
Char ACC 4	TF	95.00**	92.63***	92.63**	95.25**	94.88*	98.00	95.88	96.75*	**98.00**
	GB	97.25**	96.63**	95.00**	98.38*	97.50*	99.38	98.75*	99.38	**99.75**
Char ACC 5	TF	95.88**	94.75**	95.63**	96.25**	96.50*	97.50	96.88	97.13*	**98.25**
	GB	98.13**	97.00**	97.00**	98.83	98.25.	99.38	98.88*	99.50	**99.75**

Note: Char ACC *i* (*i* = 1, 2, 3, 4, 5) represents the character ACC obtained through the first *i* runs. The *, **and *** symbols indicate *p*-values less than 0.05, 0.01, and 0.001, respectively, after conducting the Wilcoxon signed-rank test. The maximum of each row is indicated in bold.

complexity of different network models. MACs are multiply-accumulate operations, while Params denote the count of trainable network parameters. The computational complexity of mwEEGNet was manageable.

E. Limitations and Future Works

Table III and Fig. 7 showed that some participants (e.g., P12) achieved good offline and online performance, the TF offline single trial ACC was 97.41%, while the TF online single trial ACC reached 96.78%. Similarly, the GB offline single trial ACC reached 98.50%, while the GB online single trial ACC reached 98.44%. An automated offline adaptive approach

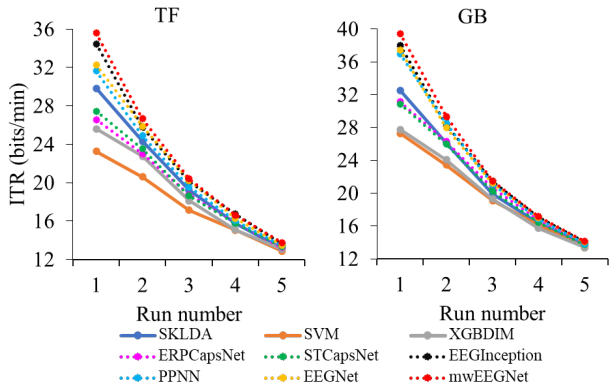

Fig. 10. The ITRs of different classifiers for the TF and GB patterns.

TABLE VI
THE COMPLEXITY MEASURE METRICS OF DIFFERENT NETWORK MODELS

Method	MACs	Params
ERPCapsNet	0.12G	7243570
STCapsNet	0.12G	7244582
EEGInception	6.73G	27618
PPNN	3.2G	12226
EEGNet	3.41G	2474
mwEEGNet	4.01G	11402

Note: The input size is (512, 1, 26, 250), MACs: the multiply-accumulate operations, Params: the number of trainable network parameters, G: Giga-10^9.

might be necessary to determine the optimal amount of training data for each participant during offline processing. For example, ERP-sensitive participants may require less offline training to obtain a suitable model. Although manually determined flash rules offer limited variability, employing the enumeration with constraints method can yield more comprehensive flash rules, albeit at the cost of increased time. While the LT2F strategy provides a straightforward means of generating flashing rules, it still has room for improvement. As shown in Fig. 2, if $m = n$, when participants perform free spelling, neighboring flashes (the double-flash problem) may occur because each flash in $Q_{\text{half A}}$ and each flash in $Q_{\text{half B}}$ have an intersection.

V. CONCLUSION

In this study, we developed a novel GB spelling paradigm that elicited more significant negative peaks in the parietal and occipital regions 200-230 ms post-stimulus onset. Our paradigm exhibited superior performance to the traditional TF spelling paradigm, highlighting the potential of the proposed growing bubble stimuli. The LT2F flash strategy also provides a more general approach to establishing flash rules, ensuring that a combination of two flashes can uniquely identify each character. We also addressed the variability in spatial features across different periods by employing a "Sub and Global" mode to enhance classification performance in the mwEEGNet model, which incorporated multi-window functionality. Our experimental results underscore the effectiveness of this approach, signifying its significance for advancing ERP-based BCIs.

REFERENCES

[1] J. R. Wolpaw *et al.*, "Brain-computer interfaces for communication and control," *Clin. Neurophysiol.*, vol. 113, no. 6, pp. 767–791, Jun. 2002.
[2] Z. A. Keirn, and J. I. Aunon, "A new mode of communication between man and his surroundings," *IEEE Trans. Biomed. Eng.*, vol. 37, no. 12, pp. 1209-14, 1990.
[3] A. Bashashati *et al.*, "A survey of signal processing algorithms in brain-computer interfaces based on electrical brain signals," *J. Neural Eng.*, vol. 4, no. 2, pp. R32-R57, 2007.
[4] J. Jin *et al.*, "MOCNN: A Multiscale Deep Convolutional Neural Network for ERP-Based Brain-Computer Interfaces," *IEEE. Trans. Cybern.*, pp. 1-12, Jul. 2024.
[5] C. C. Duncan *et al.,* "Event-related potentials in clinical research: guidelines for eliciting, recording, and quantifying mismatch negativity, P300, and N400," *Clin. Neurophysiol.*, vol. 120, no. 11, pp. 1883–1908, Nov. 2009.
[6] L. A. Farwell, and E. Donchin, "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials," *Electroencephalogr. Clin. Neurophysiol.*, vol. 70, no. 6, pp. 510–523, 1988.
[7] J. Qu *et al.*, "A novel three-dimensional P300 speller based on stereo visual stimuli," *IEEE T. Hum.-Mach. Syst.*, vol. 48, no. 4, pp. 392–399, Aug. 2018.
[8] Y. Wu *et al.*, "A spelling paradigm with an added red dot improved the P300 speller system performance," *Front. Neuroinform.*, vol. 14, p. 589169, Dec. 2020.
[9] M. Xu *et al.*, "A brain-computer interface based on miniature-event-related potentials induced by very small lateral visual stimuli," *IEEE Trans. Biomed. Eng.*, vol. 65, no. 5, pp. 1166–1175, May. 2018.
[10] T. Kaufmann *et al.*, "Flashing characters with famous faces improves ERP-based brain-computer interface performance," *J. Neural Eng.*, vol. 8, no. 5, p. 56016, 2011.
[11] J. Jin *et al.,* "The changing face of P300 BCIs: a comparison of stimulus changes in a P300 BCI involving faces, emotion, and movement," *PloS one*, vol. 7, no. 11, e49688, 2012.
[12] Q. Li *et al.*, "Optimizing the performance of the visual P300-speller through active mental tasks based on color distinction and modulation of task difficulty," *Front. Hum. Neurosci.*, vol. 13, p. 130, Apr. 2019.
[13] Z. Lu *et al.*, "The self-face paradigm improves the performance of the P300-speller system," *Front. Comput. Neurosci.*, vol. 13, p. 93, Jan. 2020.
[14] S. Li *et al.*, "Comparison of the ERP-based BCI performance among chromatic (RGB) semitransparent face patterns," *Front. Neurosci.*, vol. 14, p. 54, Jan. 2020.
[15] F. Guo *et al.*, "A brain–computer interface using motion-onset visual evoked potential," *J. Neural Eng.*, vol. 5, no. 4, pp. 477, Nov. 2008.
[16] D. Liu *et al.*, "Doubling the speed of N200 speller via dual-directional motion encoding," *IEEE Trans. Biomed. Eng.*, vol. 68, no. 1, pp. 204-213, Jun. 2020.
[17] C Guan *et al.*, "High performance P300 speller for brain-computer interface," in *IEEE Int. Workshop Biomed. Circuits Syst., Singapore, 2004*, S3/5/INV-S3/13.
[18] J. Jin *et al.,* "Optimized stimulus presentation patterns for an event-related potential EEG-based brain-computer interface," *Med. Biol. Eng. Comput.*, vol. 49, pp. 181–191, Feb. 2011.
[19] G. Pires *et al.*, "Comparison of a row-column speller vs. a novel lateral single-character speller: Assessment of BCI for severe motor disabled patients," Clinical Neurophysiology, vol. 123, no. 6, pp. 1168–1181, 2012.
[20] R. Fazel-Rezai and K. Abhari, "A region-based P300 speller for brain-computer interface," *Can. J. Electr. Comp. Eng.-Rev. Can. Genie Electr. Inform.*, vol. 34, no. 3, pp. 81–85, 2009.
[21] Z. Oralhan, "A new paradigm for region-based P300 speller in brain computer interface," *IEEE Access*, vol. 7, pp. 106618–106627, Aug. 2019.
[22] M. S. Treder *et al.*, "Gaze-independent brain-computer interfaces based on covert attention and feature attention," *J. Neural Eng.*, vol. 8, no. 6, p. 66003, Dec. 2011.
[23] D. J. Krusienski *et al.,* "A comparison of classification techniques for the P300 Speller," *J. Neural Eng.*, vol. 3, no. 4, pp. 299–305, Dec. 2006.

[24] B. Blankertz *et al.*, "Single-trial analysis and classification of ERP components—a tutorial," *NeuroImage*, vol. 56, no. 2, pp. 814–825, May. 2011.
[25] M. Kaper *et al.*, "BCI competition 2003-data set IIb: support vector machines for the P300 speller paradigm," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 6, pp. 1073–1076, Jun. 2004.
[26] A. Kabbara *et al.*, "An efficient P300-speller for Arabic letters," *Proc. Int. Conf. Adv. Biomed. Eng., Beirut*, *Lebanon*, 2015, pp. 142–145.
[27] B. Li *et al.*, "Assembling global and local spatial-temporal filters to extract discriminant information of EEG in RSVP task," *J. Neural Eng.*, vol. 20, no. 1, p. 16052, Feb. 2023.
[28] Y. LeCun *et al.*, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May. 2015.
[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015.
[30] A. Krizhevsky *et al.*, "Imagenet classification with deep convolutional neural networks," in *Commun. ACM, New York, USA, 2017*, pp. 84–90.
[31] T. Emsawas *et al.*, "Multi-Kernel Temporal and Spatial Convolution for EEG-Based Emotion Classification," *Sensors*, vol. 22, no. 21, p. 8250, Nov. 2022.
[32] H. Altaheri *et al.*, "Physics-Informed Attention Temporal Convolutional Network for EEG-Based Motor Imagery Classification," *IEEE Trans. Ind. Inform.*, vol. 19, no. 2, pp. 2249–2258, Feb. 2022.
[33] V. J. Lawhern *et al.*, "EEGNet: a compact convolutional neural network for EEG-based brain-computer interfaces," *J. Neural Eng.*, vol. 15, no. 5, p. 56013, Oct. 2018.
[34] E. Santamaria-Vazquez *et al.*, "EEG-inception: A novel deep convolutional neural network for assistive ERP-based brain-computer interfaces," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, no. 12, pp. 2773–2782, Dec. 2020.
[35] R. Ma *et al.*, "Capsule network for ERP detection in brain-computer interface," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 29, pp. 718–730, Apr. 2021.
[36] Z. Wang *et al.*, "ST-CapsNet: Linking Spatial and Temporal Attention with Capsule Network for P300 Detection Improvement," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 31, pp. 991–1000, Jan. 2023.
[37] F. Li *et al.,* "Phase preservation neural network for electroencephalo-graphy classification in rapid serial visual presentation task," *IEEE Trans. Biomed. Eng.*, vol. 69, no. 6, pp. 1931–1942, Nov. 2021.
[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015.
[39] A. Paszke *et al.,* "Pytorch: An imperative style, high-performance deep learning library," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 8026–8037, 2019.
[40] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn., 2015*, pp. 448–456.
[41] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 1251–1258.
[42] D.-A. Clevert *et al.*, "Fast and accurate deep network learning by exponential linear units (elus)," in *Proc. IEEE Conf. Learn. Repr. (ICLR)*, 2016, pp. 1–14.
[43] N. Srivastava *et al.*, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jun. 2014.
[44] J. R. Wolpaw *et al.,* "Brain-computer interface technology: a review of the first international meeting," *IEEE Trans. Rehabil. Eng.*, vol. 8, no. 2, pp. 164–173, Jun. 2000.
[45] S. P. Heinrich, "A primer on motion visual evoked potentials," *Doc. Ophthalmol.*, vol. 114, pp. 83-105, Feb. 2007.
[46] M. Ancona *et al.*, "Towards better understanding of gradient-based attribution methods for deep neural networks," *Proc. Int. Conf. Learn. Represent. (ICLR)*, *Vancouver*, *Canada*, 2018.
[47] A. Gramfort *et al.*, "MEG and EEG data analysis with MNE-Python," *Front. Neurosci.*, p. 267, Dec. 2013.