Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technical University of Munich

Human-Centered Neuroengineering: Cybathlon

Altug Yanar, Ana Ilinca Munteanu, Lukas Wogirz
✉ altug.yanar@tum.de, anailinca.munteanu@tum.de, lukas.wogirz@tum.de
July 27, 2020

Abstract — Persons with motor disorders often have very limited communication possibilities, and as such require assistive technologies. Brain-Computer Interfaces (BCIs) are a promising tool for restoring lost functions in persons with severe motor impairments, raising great hopes as they bypass muscle or nerve damage and directly translate brain signals into commands. The effective use of BCIs in communication tasks is currently burdened by the low rates of information transfer which can be achieved by using this technology. However, BCI spelling applications can be improved by designing a smart user interface and by applying text prediction. The main goal of this work was to suggest an approach and a user interface for a spelling BCI application, based on motor imagery and through techniques of natural language processing (NLP). Motor imagery signals are generated when the subject imagines moving a certain body-part. These changes in the brain signals are detected in real-time through electroencephalogram (EEG) and are then translated into commands for our spelling BCI application.
The classifiers accuracy is evaluated through the Cybathlon Brain Driver Game from ETH Zürich against a reference driver, which has an accuracy of about 60%. Our approach provided a much better accuracy, thus our driver (approx. 269.75s) has clearly defeated the gold driver (approx. 303s).
Furthermore, our novel BCI speller approach provided better results and improved the performance of the standard Hex-O-Speller approach, against three sentences, on average by 48.1%.

1 Introduction

Motor neuron diseases (MNDs) mostly affect the motor control of the muscles, by disrupting neurological connections. They include Amyotrophic Lateral Sclerosis (ALS), Locked-in Syndrome (LIS), brainstem stroke, brain or spinal cord injury, cerebral palsy, muscular dystrophies, and multiple sclerosis (MS). All of these eventually lead to functional and cognitive disabilities. As such, since they cannot speak or use sign language, these patients are unable to communicate, which negatively affects their quality of life [1].
To help these patients lead a more social life, alternative communication ways are needed. Examples of such communication systems are eye-tracking or eye-blinking spelling systems [2]. However, such systems are not suitable for patients who have no control over their ocular movements or who have uncontrollable head movements.
A technology which would enable these patients to communicate is BCI. Such a system bypasses any need of muscular movement, by directly allowing the patients to communicate through their brain waves. There are many ways of measuring the brain's activity, but the most commonly used method, is the electroencephalogram (EEG), since it is non-invasive, cheap, portable, easy to use and has a high temporal resolution [3]. After recording the brain activity in a BCI system, features of the signal are extracted and analyzed. The output is then used in the BCI application, which either replaces, or enhances the functionality of the nervous system.
One commonly studied application is the BCI Speller. Such an application allows the patient to communicate in its environment through the use of a Graphical User Interface (GUI), which shows letters, numbers or special characters. Through the brain signals recorded and analyzed by the BCI, the patient is able to select the character they want and type it on a screen. Farwell and Donchin presented the first spelling application in 1988 [4]. Patients who use BCI-spellers could regain independency and even improve their social life.
The main EEG paradigms used by the vast majority of BCI spellers are Event-Related Potentials (ERP) (P300 and Steady-State Visually Evoked Potential (SSVEP)) and Motor Imagery (MI) [5]. Motor imagery signals are generated when the subject imagines moving a limb. Leg and arm movements can be distinguished, since their signal location varies depending on which part is moved. The use of MI over SSVEP is beneficial, since it avoids the uncomfortable stimulation technique and it also can be used in severe cases of ALS, since no eye movement is necessary. This work focuses on the EEG paradigm

of motor imagery used for BCI spellers.
The performance of BCI-spellers is commonly measured by calculating the accuracy and the Information Transfer Rate (ITR) of the system. The accuracy is calculated by dividing the number of correct commands by the total number of commands. The commonly used ITR was introduced by Wolpaw in [6]. The ITR combines the accuracy and the system's speed in one variable and it is expressed as the number of error-free bits per time unit.

2 Related Work: Motor Imagery-Based BCI Speller

The basic BCI system (Fig. 1, [7]) normally has the following components: a brain activity recording device (e.g., EEG), a processor which does all the preprocessing (such as filtering and feature extraction), a decoder which classifies the extracted features into control signals for the external device (e.g., a prosthesis, a computer screen, etc.) and the external device which provides feedback for the user. The feedback is an important part of a BCI system, since it informs the user about mistakes, and motivates them towards a better performance.

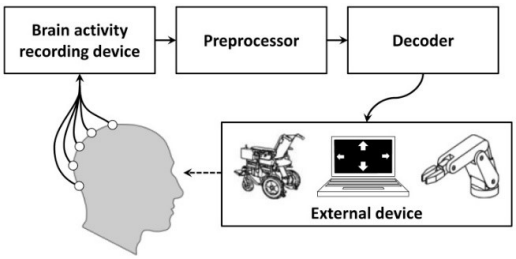

Figure 1 Components of the basic BCI system

BCIs based on motor imagery (MI) analyze the changes in brain activity in the motor cortex, while the subject is imagining the movement of a limb. As such, by detecting these changes, one can determine the intentions of the subject [8]. Different approaches have been proposed, in order to increase the detectability of MI, such as different filtering or classifying approaches, thus mainly modifying the Preprocessor and Decoder blocks of Fig. 1 [9].
Several MI-based BCI-spellers have been proposed so far. For example, D'Albis in [10] proposed a typing interface with 26 characters of the English alphabet and additionally a "space" (therefore, 27 symbols in total), which were equally divided in 3 blocks. There was also a "undo" block (thus, 4 blocks in total). The user could select one of the blocks by imagining the corresponding body part movement (in their case the right hand, left hand, both hands or both feet). When selecting one of the blocks, the 9 characters contained in it were again divided into 3 new groups, of 3 characters each, and so on. Therefore, 3 consecutive selections were needed in order to type a character.
Another MI-based BCI-speller is the Hex-o-Spell [11], which uses the motor imagery of two movements (right hand and foot) for typing 30 different characters. The characters are displayed in 6 hexagons, distributed around a circle. Each hexagon has 5 characters and a "return" slot. An arrow is placed in the center of the circle, and it is used for selecting the desired hexagons. Right hand motor imagery rotates the arrow clockwise, while foot motor imagery confirms the selection. After selecting one of the hexagons, the 5 characters and the "return" command are shown in 6 hexagons around the circle, using the same layout as in the beginning. By controlling the arrow in the same way, the subject can select the desired character, or can go back to the previous level of the user-interface, in order to correct a possible mistake.
One other recent MI-based BCI speller, is the Oct-O-Spell [12]. Its GUI is very similar to Hex-O-Spell. Additionally, the system has a switch, by which the user can turn it on and off, through a specific brain signal. Oct-O-Spell has three stages, unlike Hex-O-Spell, which only has two. The third stage was implemented in order to verify the user's intentions.
Furthermore, in 1952, Huffman published a paper [13] in which he presented a novel method for finding optimal varying-length codes for lossless data compression based on symbol frequencies. Through this, a code with minimum-redundancy can be constructed, so that the average number of coding digits per message is reduced to a minimum. Together with a binary decision tree implementation, this can be used in designing a BCI Speller. The separation according to this method is technically optimal, but we have decided to take a different approach. Based on its usability, we chose to develop our BCI Speller based on the Hex-O-Speller approach, together with a language model, namely the Natural Language Processing (NLP). Since the GUI is the first thing the user sees, we believe that user-friendliness and the general performance of the system are very important aspects, which should be considered in the development process of a BCI

Speller.
Hence, for this work we have decided to implement two different spelling interfaces, based on the Hex-O-Speller, in order to evaluate their effect on the systems performance.

3 Methods: Approaches for our Motor Imagery-Based BCI Speller

The whole BCI Speller processing pipeline is shown in Fig. 2. The task of this pipeline is to analyze a given EEG data stream, recognize left and right motor imagery, send resulting commands to a spelling GUI with the goal of writing several presented sentences. The classification challenge consists first of all in recognizing if a presented motor imagery has to be classified as right or left hand movement. To do so, the classifier has to decide for every data sample, if it corresponds to idle data (no movement imagination) or to a defined motor imagery. Accordingly, the task can be assumed as a three classes decision.

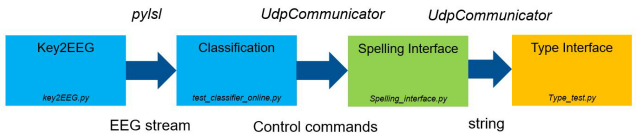

Figure 2 The BCI processing pipeline for the spelling BCI

The first block, Key2EEG, provides a continuous EEG data stream. The streamed data was recorded with a 59 electrodes EEG setup and has classified motor imagery into left or right hand movements. The state of this stream is always idle, until the right or left arrow key on the keyboard is pressed. Then, Key2EEG streams an accordingly classified sample. Otherwise it just streams recorded calibration data, which can be classified as idle.
The next block describes the classification task. The input at each timestamp is a list of 59 floats, each for one recorded EEG channel. Those lists are recorded at 100 Hz and stored in a 150 samples long buffer, which corresponds to a recording of 1.5 seconds. The output is a UDP command send via Port 5555 to the Spelling Interface block. Fig. 3 shows the inner structure of the Classification block. The first processing step of the Classification block is the preprocessing of the raw EEG data stream. It is well known that the use of the Alpha/Beta bands for motor imagery classification is more reliable than

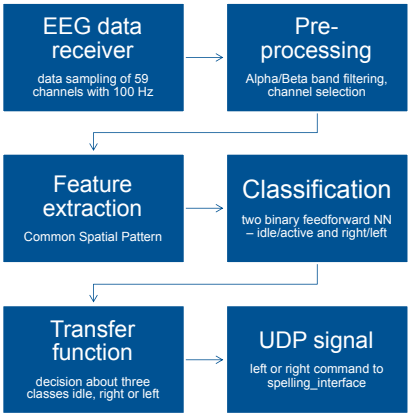

Figure 3 Process pipeline from EEG signal to right or left command

the use of all frequency bands. This ensures the elimination of most of the measurement noise, and it also filters out artifacts, such as eye blinking [14]. For a more reliable exclusion of EEG artifacts, only those EEG electrodes were chosen which are directly connected to the motor cortex. In this work the electrodes FC2, FC4, FC6, CFC2, CFC4, CFC6, C2, C4, C6, CCP2, CCP4, CCP6, CP2, CP4, CP6, FC5, FC3, FC1, CFC5, CFC3, CFC1, C5, C3, C1, CCP5, CCP3, CCP1, CP5, CP3 and CP1 were used [15]. Independent Component Analysis (ICA), which can also be used for artifact removal and channel selection was not implemented in this work, in order to have a better runtime performance.
After the frequency band filtering and channel selection, the data is fed to a feature extractor. Common Spatial Pattern (CSP) was our final choice for feature extraction, after some experiments using band power or Deep Neuronal Network (DNN) [14]. The used CSP algorithm is applicable for a binary classified data set [16].

FNN	val split size	accuracy	val accuracy
idle/active	0.1	0.8715	0.8958
right/left	0.1	0.9537	0.9167

Table 1 FNN training accuracy

The final classification is done with two Feedforward Neuronal Networks (FNN) with one hidden layer. As discussed previously in this chapter, we are faced with a three classes classification task. The used CSP feature extraction algorithm handles only two classes data sets. To handle also a three classes problem, two classifiers were used. The first FNN (I-FNN) decides whether the data samples is idle or

active and if the sample is classified as active, the second FNN (LR-FNN) classifies the sample as left or right. Table 1 shows the accuracy for a validation split size of 0.1. The results indicate that the combination of CSP and FNN works quite well in comparison to approaches like CSP and Linear Discriminant Analysis (LDA) [14]. Fig. 4 shows the processing done by the transfer function.

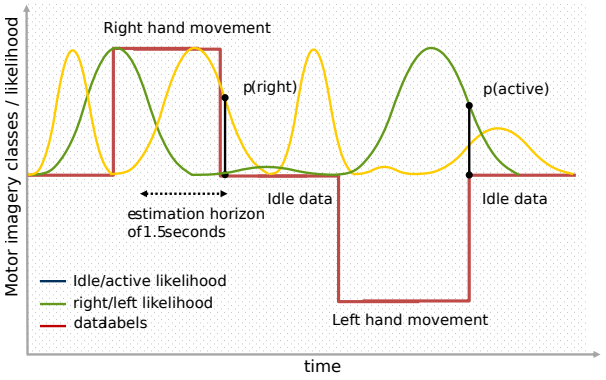

Figure 4 Schematic likelihood estimation of data samples according to real labels.

Both FNNs are classifying the received data. The I-FNN estimates the likelihood of receiving an active data set with a float between 1 and 0, similar to the LR-FNN estimating the likelihood to receive a right motor imagery. Whereas the output of the LR-FNN is a float between -1 and 1. The resulting transfer function is described in Eq. 1:

$$\hat{y} = \text{round}(a\, P_{active}\, P_{right})$$

$$\begin{aligned} &\forall\ \hat{y} \in \{-1, 0, 1\} \text{ and} \\ &\{a \in \mathbb{R} \mid 0 \leq a \leq 1\} \text{ and} \\ &\{P_{active} \in \mathbb{R} \mid 0 \leq P_{active} \leq 1\} \text{ and} \\ &\{P_{right} \in \mathbb{R} \mid -1 \leq P_{right} \leq 1\} \end{aligned} \tag{1}$$

whereby the operator round is defined as:

$$\text{round}(x) = \text{sgn}(x)\, \lfloor |x| + 0.5 \rfloor \tag{2}$$

The parameter a can be understood as a sensitivity, whether the data sample is classified as active, so right ($\hat{y} = 1$) or left ($\hat{y} = -1$), or as idle ($\hat{y} = 0$). It has to be remembered that the estimation was made out of the last 150 data samples. This leads to an effect in which the likelihood e.g. for a right motor imagery is maximized 1.5 seconds after the right arrow key was pressed. The sensitivity a has to be adjusted in a way that the increasing likelihood of a receiving right motor imagery is already recognized e.g. after 0.5 seconds. This leads to a much faster reaction time of the Classifier block. However, decreasing a also leads to more uncertain estimations. This trade-off needs to be considered.

The transfer function also has a dead time of 1.5s after each classification of a right or left motor imagery. This dead time guarantees that the classifier identifies a motor imagery only once. For instance if $\hat{y}$ oscillates around the threshold where the signal is round to a discrete output, the classifier will also oscillate between e.g. 0 for idle and 1 for right. One single received right motor imagery would accordingly results in the classification of multiple right motor imageries. Thus, with a dead time of 1.5 seconds, it is assumed that the input of an active motor imagery has to be min. 1.5 seconds apart from each other.

Fig. 5 shows the likelihood measurements of both FNNs, the transfer function output and the true label with $a = 0.7$ during the stream of only idle EEG data. It can be clearly seen that a is too sensitive, hence some wrongful classifications occur.

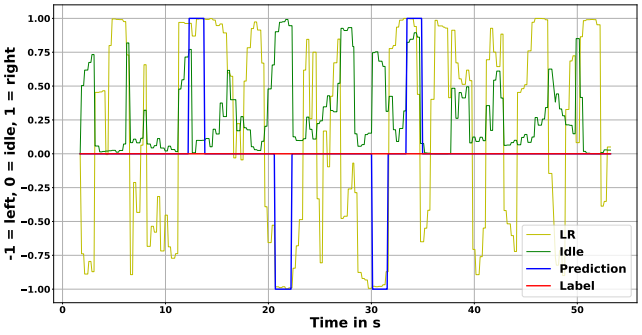

Figure 5 Likelihood estimation over 50 seconds idle EEG data with $a = 0.7$

Fig. 6 shows a classification of an offline assembled data set with known labels. After several attempts with different values of a, the best trade-off was achieved with $a = 0.7$. Additionally, the defined dead time of 1.5 seconds turned out to be sufficient. Especially because, after a first correct classification, the LR-FNN tends during the next 1.5s to then classify it wrongly. This effect can clearly be seen in Fig. 6 around second 15. Based on the previous assumptions and decisions, a Classification block was created which classifies with a very high accuracy, but in turn requires more time.

Finally the Classification block sends a right UDP command, when $\hat{y} = 1$, and a left UDP command, when $\hat{y} = -1$. If an idle state is classified, no com-

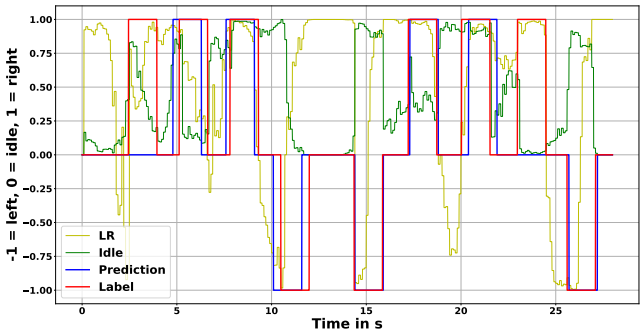

Figure 6 Likelihood estimation over 50 seconds idle/right/left EEG data with $a = 0.7$

mand is sent.

The whole pipeline in the Classification block needs about 70 to 80 ms [1]. To keep the system working reliably, the classification process is done with 10 Hz. According to this classification rate, it is ensured that the process has a small buffer, in order to avoid any information loss or sample buffer overflow.

The UDP commands are received by the Spelling Interface block and are converted into control commands for the GUI of our BCI Speller.

As already mentioned, instead of designing completely different BCI speller approaches, the idea in this work was to use the Hex-o-Spell method and make a comparison between two different setups. From our perspective, the Hex-o-Speller is a fast, easy and overall a very user-friendly interface. The first implemented setup of the Hex-o-Speller keeps its basic structure, as described in [11]. The only slight changes were that we used six circles instead of six hexagons and we replaced the arrow by displaying the current circle via a different color. Fig. 7 shows the resulting spelling interface of the first setup.

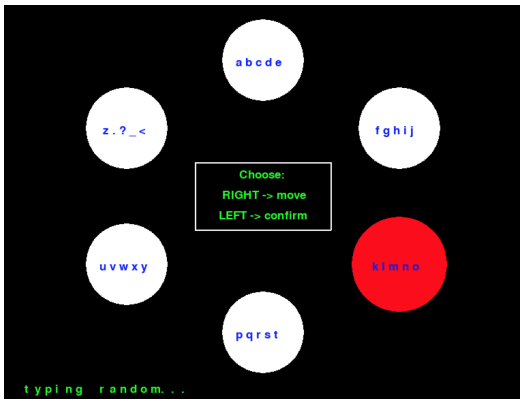

Figure 7 First setup of the Hex-o-Spell spelling interface

[1] System specifications: CPU i7-1065G7, 16 GB RAM

Right classified motor imageries rotate clockwise between the circles, where the red color is indicating the current circle. The selection process is done by left motor imageries choosing the desired circle and activate stage two (Fig. 8) in which now the 5 characters and a "return" command (empty circle) are displayed. Finally, a second selection command finishes the process and picks the desired character. One important aspect to be considered is the fact that the BCI classification is not optimal and therefore, wrong classifications are always to be expected. Furthermore, the selection of the wrong circle represents a highly possible scenario. For this reason, it is necessary to provide the user with the required return and/or redo options. In this case, the selection of the empty circle lets the user return from being in the wrong stage 2 and the '<' character can be used for deleting the last typed character.

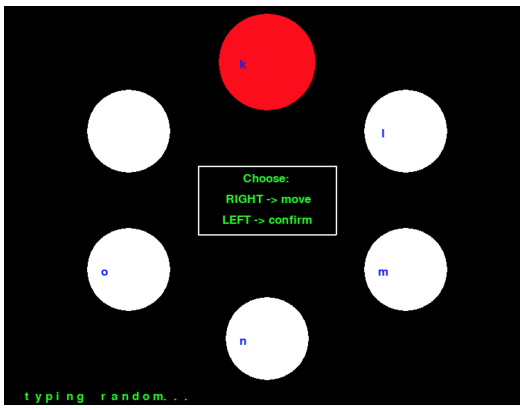

Figure 8 Stage 2 of the spelling interface after selecting the desired circle

The implementation of the character selection process is realized through a 2x6 state matrix where the rows are equal to stage 1 and 2 and the columns are representing the circles. For each selection via 'left' command a '1' is written to the current row so that, at the end in each row there is exactly one '1' and the rest is '0'. Depending on the position of the 1's, each combination corresponds to one character. Eq. 3 shows the representation of the character 'n' in the state matrix M.

$$M = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{3}$$

The Classification block is primarily limited in speed. Therefore, it would be an improvement to reduce the necessary amount of control commands needed to type one character, as well as to reduce the amount of characters necessary for recognizing the desired

word. For this, one approach would be to reorder the characters. The initial order of the 30 used characters is the alphabetic order, with the special characters at the end. To bring the characters in a much more efficient order, a Natural Language Processing algorithm (NLP) can be used. The NLP algorithm estimates from a given input (meaning the last selected characters), what character is most likely to follow. The NLP which is used in this work is a Recurrent Neuronal Network (RNN) with a Long short-term memory (LSTM) layer and Embedding for feature extraction [17]. It was trained on the book "Pride and Prejudice" written by Jane Austen, which contains 768934 characters [18]. The NLP is capable of predicting the likelihood of being the next one after the input string, for each character , as well as of predicting a whole word depending on the first given characters. This leads to the performance of selecting the desired character, in the best case scenario, with only two commands. Because the Hex-O-Speller provides the user with 30 slots for letters or special characters, the '?' character from variant 1 is cancelled, so one slot is still available. This slot is shown as first in the Hex-O-Spell interface order and is filled with the predicted word as shown in Fig. 9. This makes it possible to use auto-completion.

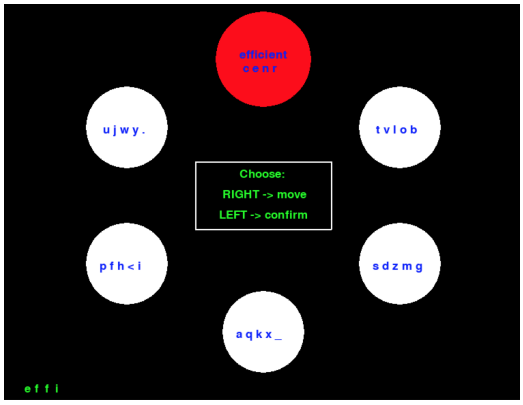

Figure 9 Second setup of the Hex-o-Spell spelling interface with auto-completion and efficient ordering

It needs to be mentioned, that the Classification block is not always accurate. Techniques to handle wrongly classified motor imageries and subsequent wrong command controls such as 'return' and 'delete' were previously described. Meanwhile, there are further strategies for correcting or preventing errors. One of these strategies is a spell checker. In the cases where it was not possible to avoid misspelling through the basic correction strategies, the spell checker can correct a slightly misspelled word (Fig. 10) [19]. This is a further advantage of the second setup.

Figure 10 Spellchecker extension of the second setup of the Hex-o-Spell spelling interface correcting the misspelled word 'possible'

In summary, the Spelling Interface block receives a sequence of two different UDP commands. Those commands are noisy, according to the quality of the Classification block. The sequence is decoded by the Spelling GUI into strings that correspond to desired words, which are finally sent to the Type Interface block. To reduce errors, the GUI provides the possibility to undo or return commands. The NLP adds auto-completion and character to help prediction to reduce the amount of commands necessary for typing a word. Finally, the spell checker adds an auto-correction to the complete word, which represents the output of the Spelling Interface. After this, the word is sent via UDP to the Type Interface block and is then compared with the desired sentence.

4 Experiment and Results

We evaluated our BCI classification approach in two different environments. First, we used the EEG classifier in the Cybathlon Brain Driver Game, from ETH Zürich. We competed against a provided reference driver (gold driver) and the result was that our driver clearly defeated the reference driver. On average, we achieved a lap time of about 270s, while the gold driver achieved a lap time of only 300s. The gold driver had a known accuracy of about 60%. Second, we evaluated both Hex-o-Spell approaches against three randomly selected sentences, as shown in Table 2.

Nr.	Sentences
1	War does not bring anything good to the common people.
2	The prospect of the Netherfields ball was extremely agreeable.
3	The communication excited many professions of concern.

Table 2 Randomly chosen sentences for evaluation

The results in Table 3 show the number of characters typed in 100s, as well as the value char/min, for each of the 3 sentences and each of the two setups.

	Sentence Nr.	**Setup 1**	**Setup 2**
characters	1	33	47
char/min	1	19.8	28.2
characters	2	32	52
char/min	2	19.2	31.2
characters	3	33	46
char/min	3	19.8	27.6

Table 3 Results of each Hex-o-Spell setup against the three sentences

The results show that the extended Hex-o-Spell (setup 2) clearly outperforms the basic structure (setup 1). This leads to the conclusion, that the chosen additional features were appropriate and highly reasonable. The second setup improves the char/min performance on average by 48.1% .
Accuracy, error rate and the information transfer rate could not be properly evaluated on the complete pipeline, due to problems between the communication of *test online* and *spelling interface*. The development environment, like the UDP communicator, as well as the interface between this communicator and the spelling interface had several issues, which could be fixed for the most part. The remaining problem of a non-performing and asynchronous behaviour of the spelling interface, could not be fixed in the given time frame. It can be observed that the spelling interface block tends to receive the plotting of the GUI or the receiving of UDP data in a single thread. It seems that it is not possible to do both in one loop. This results in a loss of UDP commands or a proper realization of the GUI, depending on the given UDP commands. Finally, this leads to an extremely slow experience, which, from a usability point of view, is not feasible.

5 Discussion

Our results showed, that especially in terms of usability, the efficient reordering of the characters is definitely a helpful feature which enormously improves the typing speed. Furthermore, for the case of long words, the auto-correction also represents an important add-on. Also, particularly for the first setup, after typing the first characters, the monotonous selection process becomes annoying and also exhausting, in particular for longer words. For this reason, in our opinion the use of NLP extensions in future BCI spellers will play an important role. Unfortunately, the spell checker could not be evaluated properly since we just considered the keypress scenario, without EEG. But, regarding the not optimal classification accuracy, the amount of possible errors is high and therefore, spell checking is an easy extension which could compensate for the lack of accuracy.
A possible future improvement could be a query for each written character or word, to control the correctness of typing. Currently, in the ideal case, it needs ten clicks to reach the 'delete' character, which slows down the error correction process. One approach which could be used in the future, is to include the Error-Related Components, such as the Feedback-Related Negativity, in our processing, since they have a high potential to improve the human-machine interaction. These signals occur when the subject receives feedback after committing an error [20]. In our case, this would happen if a wrong letter is selected. The letter will be shown on the display, the subject will see it is incorrect, which will generate an FRN. In our online processing, when the system would detect an FRN, it could for example correct the error automatically, by deleting the wrongly typed letter. This could potentially eliminate a further trial of redoing a typing action (deleting), which is time consuming and requires high concentration.
Our MI-based BCI-speller is a gaze-independent speller, which, as previously mentioned, might prove useful even for patients with advanced motor impairments. Since the speller is motor imagery based, the user can gain control over the system through practice. The system's spelling speed was also dependent on the user interface used. Our system benefits from the advantages offered by the use of MI, such as gaze-independency and no external stimulation necessary. Nevertheless, the disadvantages of an MI system, such as the high complexity of the data analysis and the long training periods, need to be mentioned.
The future of such systems is promising, but some gaps, such as paying more attention to the GUI design and involvement of the patients in the whole design process, need to be filled in order to achieve competent BCI spellers.

6 Social and Ethical Relevance

Responsible Research and Innovation (RRI) presents a new approach which describes the relation between science and society [21]. RRI offers three different perspectives on responsibility: a. Anticipation: What are potential risks? What don't we know? b. Inclusion: Beneficial for whom? c. Responsiveness: How do we know we are right? All of these dimensions need to be taken into consideration when designing BCIs. The researchers must consider and include the end-user in their approach and processes. The users or subjects should be asked if they want to be part of the whole developmental process, or if they just want to be concerned with the end-product. This will give them decisional power. By giving back the power to the user, their participation is enabled and true inclusion is achieved. One must consider "disabilities" as simply "different abilities" and should work with them towards a better product. In this way, the users will feel like they are in control, and this will give them a feeling of agency.

Yet another important issue which arises when designing BCIs is the privacy issue. Since brain data is acquired and it is used in order to "decode" what the subject's thoughts or intentions are, privacy concerns are raised. BCIs implicate physical, informational, and decisional privacy [14]. Although the promise of a BCI which could allow a locked-in person to speak, or a tetraplegic move a robotic limb is very compelling, privacy concerns must not be treated as trivial.

References

[1] H. Hanagasi et al., "Cognitive impairment in amyotrophic lateral sclerosis: Evidence from neuropsychological investigation and event-related potentials," *Brain research. Cognitive brain research*, vol. 14, pp. 234–44, 09 2002.

[2] P. Majaranta and K.-J. Räihä, "Twenty years of eye typing," 01 2002, p. 15.

[3] J. Wolpaw et al., "Wolpaw, j.r.: Brain-computer interfaces for communication and control. 'clin. neurophysio1. 113, 767-791," *Clinical neurophysiology : official journal of the International Federation of Clinical Neurophysiology*, vol. 113, pp. 767–91, 07 2002.

[4] L. Farwell and E. Donchin, "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials," *Electroencephalography and clinical neurophysiology*, vol. 70, pp. 510–23, 01 1989.

[5] A. Rezeika et al., "Brain–computer interface spellers: A review," *Brain Sciences*, vol. 8, p. 57, 03 2018.

[6] J. Wolpaw et al., "Brain-computer interface technology: A review of the first international meeting," *IEEE transactions on rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society*, vol. 8, pp. 164–73, 07 2000.

[7] A. Mora-Cortes et al., "Language model applications to spelling with brain-computer interfaces," *Sensors (Basel, Switzerland)*, vol. 14, pp. 5967–93, 04 2014.

[8] G. Pfurtscheller and C. Neuper, "Neuper, c.: Motor imagery and direct brain-computer communication. proc. ieee 82(7), 1123-1134," *Proceedings of the IEEE*, vol. 89, pp. 1123 – 1134, 08 2001.

[9] G. Dornhege et al., *Toward Brain-Computer Interfacing*, 01 2007.

[10] T. D'Albis et al., "A predictive speller controlled by a brain-computer interface based on motor imagery," *ACM Transactions on Computer-Human Interaction*, vol. 19, pp. 20:1–20:25, 10 2012.

[11] B. Blankertz et al., "The berlin brain-computer interface presents the novel mental typewriter hex-o-spell," *Clinical Neurophysiology*, vol. 113, 01 2006.

[12] L. Cao et al., "A synchronous motor imagery based neural physiological paradigm for brain computer interface speller," *Frontiers in human neuroscience*, vol. 11, p. 274, 2017.

[13] D. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, pp. 1098 – 1101, 10 1952.

[14] C. Nam et al., *Brain-Computer Interfaces Handbook: Technological and Theoretical Advance*, 01 2018.

[15] S. Sr et al., “Classification of motor imagery based eeg signals using sparsity approach,” 12 2017, pp. 47–59.

[16] C. Brunner and et al., “mne.decoding.csp: Eeg signal decomposition using the Common Spatial Patterns (CSP),” 2019, software available from https://mne.tools/stable/generated/mne.decoding.CSP.html. [Online]. Available: https://github.com/mne-tools/mne-python/blob/maint/0.20/mne/decoding/csp.py#L22-L559

[17] C. Meyer and et al., “TensorFlow: Text generation with an rnn,” 2020, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/tutorials/text/text_generation

[18] J. Austen, “Pride and Prejudice,” 1813, urbana, Illinois: Project Gutenberg. Retrieved July 7, 2020,. [Online]. Available: http://www.gutenberg.org/ebooks/1342

[19] T. Barrus, “pyspellchecker: Pure python spell checker,” 2020, software available from https://pypi.org/project/pyspellchecker/. [Online]. Available: https://github.com/barrust/pyspellchecker

[20] J. d. R. Millan, “You are wrong!—automatic detection of interaction errors from brain waves,” 01 2005, pp. 1413–1418.

[21] M. Sabine and S. Dickel, *Partizipation, Responsivität, Nachhaltigkeit*, 09 2016, pp. 225–242.