Improving BCI Performance *after* Classification

Danny Plass-Oude Bos
HMI, Faculty of EEMCS
University of Twente
Enschede, The Netherlands
d.plass@utwente.nl

Hayrettin Gürkök
HMI, Faculty of EEMCS
University of Twente
Enschede, The Netherlands
h.gurkok@utwente.nl

Boris Reuderink
Donders Centre
for Cognitive Neuroimaging
Nijmegen, The Netherlands
b.reuderink@donders.ru.nl

Mannes Poel
HMI, Faculty of EEMCS
University of Twente
Enschede, The Netherlands
m.poel@utwente.nl

ABSTRACT

Brain-computer interfaces offer a valuable input modality, which unfortunately comes also with a high degree of uncertainty. There are simple methods to improve detection accuracy after the incoming brain activity has already been classified, which can be divided into (1) gathering additional evidence from other sources of information, and (2) transforming the unstable classification results to be more easy to control. The methods described are easy to implement, but it is essential to apply them in the right way. This paper provides an overview of the different techniques, showing where to apply them and comparing the effects. Detection accuracy is important, but there are trade-offs to consider. Future research should investigate the effectiveness of these methods in their context of use, as well as the optimal settings to obtain the right balance between functionality and meeting the user's expectations for maximum acceptance.

Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Input Devices and Strategies

Keywords

Brain-computer interface, multimodal, context-aware, evidence accumulation, user expectations.

1. INTRODUCTION

Brain-computer interfaces (BCIs) do not function as accurately as the keyboard or mouse. This is inherent in the type of signal, which is also why speech recognition and eye movement detection are not perfect, although they have improved a lot over the years [5, 14]. The sensors easily pick up noise. It is very difficult to distinguish between activity intended for control, and other activity. Moreover, there is the challenge of robustness to change in the environment, change in the user, and over different users. These problems are reflected in the way most BCI experiments are set up: the user is preferably put in a shielded room; is instructed to not move or blink, and stay relaxed; and is doing a very simple task.

Is this lack of accuracy of BCIs a problem? BCIs offer a unique input that is private, hands-free, and possibly closer to the user's intent than any other input modality [30]. But the detection accuracy and, related to this, the amount of effort involved, are very important to the user [29]. Users prefer interfaces with allow a high level of control with little effort. Besides, there are some consequences of the current situation. Those limited lab experiments mentioned in the previous paragraph are difficult to translate to the real-world situation, where the user will not be so restricted in context nor behaviour. Additionally, the variety of both mental tasks or states, and their interpretations, are severely limited.

Fortunately, even after the detection phase, there are a lot of ways to boost performance, and make the signal more usable for control[1]. This paper discusses a number of them, divided into two categories: gathering additional, external evidence, and transforming the obtained classification results by themselves. In the end, however, not only the functional properties of the control signal are important, but also how well it matches user expectations. The conclusions provide some recommendations for future research.

2. THE ONLINE BCI CYCLE

To be able discuss when and how to apply the methods that we will describe, first, we need a model of the online BCI cycle. Figure 1 details the different modules, or phases, in the system, and the data streams in between. This model is largely based on the framework by Mason and Birch [19], with some small adjustments which will be discussed.

[1]With control is not only meant intentional control by the user, but also unintentional input based on passive observation of the user. In both cases, the system responds to the information gathered from the user. For simplicity, even in the passive situation, we will therefore talk about 'control'. In the passive situation, there are still correct and incorrect interpretations of the user's mental state by the system, and if there is feedback to the user in some way, there are still user expectations that could be met.

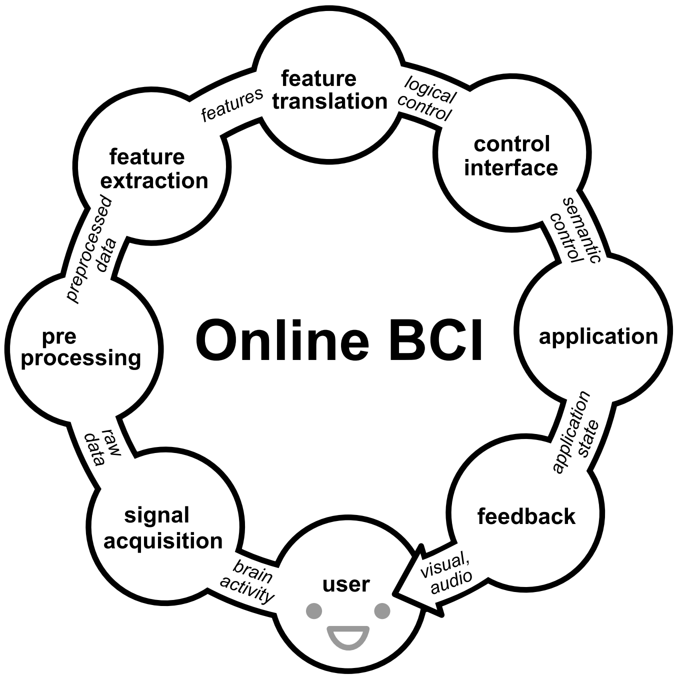

Figure 1: Model of the online BCI cycle, including the data streams between the processes.

The *user* either consciously executes some mental task, or is passively monitored for specific mental states. This brain activity is *acquired* through some brain activity measurement method, such as electroencephalography (EEG, which would consist of electrodes, an amplifier, and likely some additional software by the hardware manufacturer). Here we combined the *electrodes* and *amplifier* modules from Mason and Birch together in one generic module labeled *signal acquisition*.

The raw brain data is *preprocessed*, mainly to improve the signal-to-noise ratio by attenuating the effects of artefacts and noise. Mason and Birch combined this step with *feature extraction*. Many researchers, however, prefer to address this function separately, as is also shown in the model by Van Gerven, et al. [40]. Therefore we also gave it its own module. From this preprocessed data, *features* are then *extracted* that provide information on what the system is supposed to detect.

Feature translation is not a term that is commonly used in this context, but this generic word from the Mason and Birch model appropriately covers the whole range of classification and regression. The results of this module can be hard class labels (e.g. LEFT), soft class labels (80% certainty of LEFT), or some other values that have logical meaning (level of concentration of 0.4)[2].

The *control interface* transforms these logical inputs into semantic control signals that have meaning within the context of the application. This module is often not represented in other models, but Mason and Birch did see the importance of this particular function. It is also particularly relevant to this paper, as many of the proposed methods would be applied in this phase. The resulting semantic control can either be discrete or continuous (for example, key and button presses vs. mouse position and joystick).

On a side note, when both the recognition software (up to and including feature translation) and the application are proprietary, the control interface may be the only way to tie them together in the desired way.

At this point in the pipeline, we deviate from the Mason and Birch model, because of a difference in the context of use of the BCI. Their model reflected the use of BCI to control physical devices, where our model is specified for control of software applications. The functions of Mason and Birch's device controller and the device itself are merged within the application itself (possibly internally modelled according to the model-view-controller design pattern, well explained in [8])

The *feedback* module is simplified here as solely based on data coming from the application. The feedback can however also show representations of output from other modules. Any feedback that helps the user form a suitable mental model of the system could be helpful in improving the level or at least the sense of control. In most cases, the feedback (view) on the application state is handled by the application itself. If the application is adjustable, for example through add-ons, the feedback from other modules can be integrated. If not, it will be necessary to create a separate software application solely to provide this extra information to the user, for example, with overlays on top of the view of the original application.

To conclude, the main modules in which to adjust the control, after classification (or, the more generic: feature translation), are the control interface, and the application.

3. GATHERING EXTERNAL EVIDENCE

Now we have explained the BCI cycle in which the following methods will be used, it is time to give an in-depth explanation of each method, starting with the ones based gathering external evidence, such as from other modalities and the context.

3.1 Multimodality

Multimodal systems process multiple combined user input modalities in a coordinated manner [24]. Multimodality can help in handling errors, improving task performance and broadening the user spectrum with BCI systems [10]. In the BCI community, multimodal BCI systems are referred to as hybrid BCIs [27]. A hybrid BCI not only combines brain data from other modalities, but may also combine different brain responses. For example, a hybrid BCI speller can rely on the P300 response for spelling letters and on the steady-state visually evoked potential (SSVEP) response to switch the spelling on and off [25].

The coordination of the various inputs can be categorized according to two aspects: the treatment of the multimodal data and the temporal relation of the modalities [22]. The multimodal data streams can either be treated *independently*, or they can be *fused*. Fusion can be done at data-, feature- and decision-level, also called early, intermediate, and late fusion. Data-level fusion is performed

[2]Another interesting option, but outside of the scope of this paper to discuss in detail, is to have not only probability information, or strength, but both. In that case, the feature translation could provide information such as 'a level of concentration of 0.9 with a probability of 2%' which can then be translated into an application command that has less effect than if the probability would be higher.

when integrating observations of the same kind, such as the electrical data obtained from EEG and EOG. Feature-level fusion is for modalities which are tightly synchronised, such as frontal EEG and facial audio-visual data. Decision-level (or classifier) fusion, is used for modalities that are not so tightly coupled, such as EEG data and speech or keyboard input. There are also two ways of temporal coordination: the processing can either be sequential or parallel. In either case, the treatment can again be independent, or fused.

For the goal of this paper, the most applicable method is parallel, decision-level fusion, as it is about combining after-classification results from different inputs that happened simultaneously, to obtain a more accurate decision. One of the easier methods to do this, is through majority voting [17], in which the combined result is the one that most classifiers agreed upon. For more complex methods, see [12, 23].

This method could be implemented in the control interface, where it functionally seems to belong. If the application is unadaptable, the control interface is the only option. In case the application also needs to show a response to each input modality by itself, it may be better to implement the fusion in the application directly.

3.2 Context

Another potential source of information is the context, which can be determined from additional sensors, or from the system or application itself. According to Abowd *et al.*: "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves" [1]. Concrete examples of context are: time of day, location, movement, recent interaction history, the user's emotional state, and the user's focus of attention.

Reducing action space.

Context information can be used to determine what actions the user is probably going to do, therefore reducing the likely action space, which improves the accuracy of the input interpretation. After spelling 'TH', the classifier might return high values for both 'S' and 'E'. Based on the recent actions, in combination with a *language model*, it is easy to determine that 'E' is more likely[3]. Two BCI examples that make use of the predictive properties of text are Dasher [7], and Williamson's Hex-O-Spell [41]. The dynamics of the interaction reflect the level of certainty: it is easier to select more likely options, and more effort is required to select a less likely one. This results in increased accuracy and efficiency.

Similar concepts are *snapping* (when close to a likely target, automatically go to it), and *hysteresis* (the dynamic for changing states depends on what state you come from, to prevent undesirable oscillations from one state to the other; this is also known as debouncing). An example BCI that uses hysteresis is αWoW. In this adaptation of the role-playing game World of Warcraft®, the player's avatar is an elf when relaxed, but changes into a bear when agitated [38]. The relaxation level is based on a transformed level of parietal brain activity in the alpha band. To change to bear, the transformed alpha level needs to be below 0.3, and to change to elf above 0.7. In terms of dynamics based on certainty, this means that once the user is in one state, she is less likely to change to the other. The larger the distance between the two thresholds, the less likely it is that state changes will occur in general.

If certain actions always occur together (the likelihood of the next action is 100%), they can be combined in a *macro*. Back to the case of a P300 speller, instead of spelling single characters, the user now selects a phoneme, word, or whole sentence with one command. Scherer *et al.* use macros to execute more complex and flexible commands [34]. This is also related to the notion of process control (one step forward) versus goal selection (go to the kitchen) [42, 33].

When the implementation of these methods requires knowledge from the application, it is best to implement it in that module. For example, in the case of a speller application, the control interface should not model the effect of the backspace action that deletes the previously-typed character. This is the responsibility of the application. So the knowledge as to what text has actually been spelt so far is only available from the application module. Snapping also requires knowledge about where possible targets are in the application to be able to snap to them. Hysteresis may be implemented in the control interface, as it only requires the knowledge that the actions that belong to the extreme states are opposites and can never occur simultaneously. This logic could be imposed from the mental task or state (i.e. the user cannot be stressed and relaxed at the same time), or by the response from the application (an avatar cannot be a bear and an elf at the same time).

Increasing action space.

Similarly, context can provide the user with a large range of system commands, controlled through a limited set of input actions. Scherer *et al.* give an example of this in World of Warcraft®: when the player is close to a dead enemy, she will loot the corpse. Is the player close to a quest giver, she will talk to him [34]. To read more on how to implement context-aware systems, we recommend the survey by Baldauf *et al.*, which discusses various approaches used in existing systems and presents a design framework [2]. This method obviously requires application knowledge, so that is where it has to be implemented.

Even if context is not used explicitly, it will affect the user's brain activity. It may cause non-stationarity in the signals, which may need to addressed to maintain a good performance [31]. Context could also improve BCI performance, for example, if the situation is more motivating [15]. And perhaps the difference in brain activity can be used as a source of context information by itself.

4. TRANSFORMING CLASSIFICATION

The unsteady feature translation (classification, regression) results can also be transformed by themselves, to increase the certainty of correct detection, and make the characteristics of control better match the expectations of the user given the application. Although the methods below can be adjusted to 'hard' class labels – by giving them a 'soft' probability level of 100% – having access to actual probability information yields more accurate results. Most methods are also applicable to continuous output, for example produced by regression models, such as for the workload or stress level of the user.

[3]More on the redundancy in the English language in [35].

4.1 Accumulating Evidence

A common method to increase BCI detection accuracy is *averaging* over multiple classifications[4]. In Bacteria Hunt, over half of the classifications needs to return true, to eat the target [20]. Averaging smooths the classification results, making it less sensitive to noise. The *moving average* algorithm multiplies the most recent classifications with a smoothing vector, which determines the amount to which newer and older samples contribute to the final result. In αWoW, this simple technique reduces the effect of outliers and provides more smooth interaction [38]. *Smoothing* will slow down the interaction. To reduce this effect, one can use exponential smoothing [11, 26], or IIR filters that have a quick response. This method can be implemented in either the control interface or the application, in which case the control interface is the preferred place for this function.

Similar to BCIs, eye movement data is jittery, noisy, and uncertain. Besides, it is very difficult to distinguish intent from 'normal activity'. Jacob has proposed a solution to this so-called *Midas touch* problem by introducing a *dwell time*: looking at a target for a certain time to select it [13]. In the case of BCIs, the user has to stay in a certain value range for the given duration, to trigger a specific command. This method has been used to select a target with a BCI-controlled cursor [6], to confirm actions in BCI-controlled videogames [34, 38], and to improve classification results in general [36]. From eye movement-based control it is known that the exact duration of dwell time is critical to the flow of the interaction[13]. For brain-computer interaction, the optimal duration is likely to be different and also dependent on the mental tasks used for control. Like most methods mentioned in this paper, dwelling will increase the time to give the command. If application knowledge is required – for example, to know whether the user is hovering over a target – the appropriate place to implement this method is the application.

4.2 Refractory Period

Refractory periods suppress the BCI for a while after a command has been triggered, to reduce the number of false positives [36]. In video games, this is referred to as 'cooldown'. Pfurtscheller *et al.* used it to prevent a device, that enabled a paralyzed patient to grasp a cylinder with his hand, from switching grasping states too quickly [28]. As Bashashati *et al.* rightly observe: "As the debounce period is increased, the false activation rate is decreased for a given true positive rate. However, with increasing this time period, the reactivation time of the BCI system is impacted. The trade-off is clear and one needs to consider this for a given application" [3]. This method needs to know when an action has been triggered as a result of the control signal. It could be implemented in the control interface, but as it is the application that decides what the actual system response is to the input, that might be a better location.

4.3 Class Balancing and Normalization

When the output is biased towards one class, this can occur for a variety of reasons, such as changes in the mental state of the user [4, 31]. Although most solutions to this are applied before classification, it can also be addressed afterwards, by applying normalization methods that balance the output. Adaptive z-score normalization automatically adjusts the system to the user in αWoW. It balances the output with respect to its mean and standard deviation, which avoids a bias towards a specific mental state, and ensures that the user does not get stuck [38].

There is an interesting side effect of adaptive z-score normalization. If the running history contains many samples, as expected, the result is similar to the input, except for the scale (assuming a normal distribution). If the running history is relatively short, however, the transform becomes highly sensitive to changes in the control signal.

A possible danger of class balancing is over-accommodation. The actual class ratios should be investigated in practice. Also, when balancing is done too aggressively, it can make it very difficult for the user to generate brain activity that is extreme enough to trigger the intended action.

This method is best implemented in the control interface. This is functionally the most suitable place, and no semantic knowledge is required.

[4]This assumes the classifications are to some degree independent of each other; otherwise accumulating more predictions does not add new knowledge. This independence is, for example, affected by the overlap of sliding windows.

5. HOW TO CHOOSE

Which method is most suitable in a specific situation mainly depends on two things: (1) Location: what information you can access, and which modules you can modify; and (2) Effects: how to better meet the functional requirements of the application, as well as the user expectations. Table 1 gives an overview the different methods, where to apply them, and how they affect the control signal.

5.1 Location

As we have shown, there are two places to implement these methods after classification or regression: the control interface, and the application. The control interface provides discrete, hard class labels, or continuous values, being soft classification predictions or regression values. It can also maintain a history of these logical control inputs, and the semantic control outputs it has generated. The application has knowledge of its own state, and the semantic control inputs it has received. As both modules have knowledge of the semantic control signal, some methods can be implemented in either. In such a case it is often preferable to put the function in the control interface. Another reason to decide on a particular module is if the other module is simply not modifiable.

5.2 Effects

Each method has a number of effects, and side effects. When applying these methods in order to improve the level of control over the input, one has to take into account the requirements of the application and the expectations of the user.

Bitrates.

Common sacrifices to gain detection accuracy are response time (as the time from action to detection is increased) and bandwidth (as the total time for each detection is increased), both aspects of speed. Similarly, methods that increase robustness inherently decrease sensitivity. To obtain the high-

Method	Location	Effects
Multimodal fusion	CI, App	Higher accuracy, if the other inputs provide new, useful information
(Language) model	CI, App	Higher accuracy, if good model
Snapping	App	Higher efficiency; lower sensitivity
Hysteresis	CI, App	Higher efficiency; lower sensitivity
Macros	CI, App	Higher efficiency; more restrictive
Context-dependent response	App	Higher accuracy if that one input is well-recognized
Smoothing, averaging, dwell time	CI, App	Lower sensitivity; longer response time
Refractory period	App, CI	Lower FP, but possibly also less TP
Normalization, balancing	CI, App	Decreased bias; generalizability over users; adjusted sensitivity

Table 1: Post-processing method comparison. Location: CI = control interface, App = application. Preferred location listed first. FP = false positives (false alarms). Lower sensitivity (also lower FP) implies a debouncing effect and an increase in robustness.

est bitrate[5] [16], the trade-off between accuracy and time per detection needs to be optimized.

Related to bandwidth is the cost of different types of errors. In a first person shooter, if ammunition is abundant, it is best to shoot as often as possible, increasing the chance of hitting the target. False negatives, not shooting when the user tries to, are unwanted. On the other hand, if ammunition is scarce, it is costly to waste bullets, so it is better not to shoot. False positives, shooting when the user does not want to, are undesirable. Thus it is not only the recognition of the BCI that affects the throughput of information. The application also determines how costly it is to make an error and how easy it is to correct one. This provides another angle for improvement. This is also an important consideration in the case of passive interaction. If the system affects the background music, the cost is at worst some annoyance by the user. If the system adjusts the user interface (e.g. to the user's workload), it has to be carefully designed, as errors in detection may directly affect the usability.

Besides the fact that it can be better (or worse) to do nothing than to do the wrong thing, the user will not always be providing actionable input. Adding an idle state, and rejecting classifications with too low confidence levels, may help in increasing the bandwidth and having the interaction better match user expectations [18, 32].

Expectations.

The acceptance of BCI technology does not only depend on its functional properties, but also on how well it matches user expectations. As Jacob mentions "[The human-computer dialogue] should correspond to what the user thinks he or she is doing, rather than what his eye muscles are actually doing" [13]. This is also applicable to brain activity-based interaction. These expectations are highly dependent on the application. When controlling a race car in a racing game, the user will expect fast response times and high accuracies. The situation changes when the input is used to control a hamster. Perfect control may even be unwanted, as it is more interesting if the animal appears to have a will of its own [39]. To address the speed and accuracy issues of BCIs, perhaps controlling a snail is best.

To evaluate how well a particular BCI system as a whole meets expectations[6], the SUXES method is very suitable [37]; particularly the adjusted version for BCI by Gürkök *et al.* [9]. This method consists of two questionnaires, which contain both pragmatic (such as speed and accuracy) and hedonic items (such as naturalness and enjoyability). Each item is rated on a 7-point scale with opposite word pairs at each end (slow – fast). After an introduction to the application, the user indicates a zone of expectations in the first questionnaire. After using the application, the user marks the actual experience in second questionnaire. This makes it easy to compare experiences and expectations.

6. CONCLUSIONS

In a thorough survey of BCI signal processing algorithms of over 200 papers, Bashashati *et al.* observe that "30 BCI designs ... use post-processing algorithms to reduce the amount of error in the output of the BCI system" [3]. This means that, in 2007, for more than 85% of BCI systems, there could still a lot be gained in terms of performance. Although the numbers will have improved somewhat since then, this post-processing is still an under-discussed topic in the field, and deserves more attention.

We described a collection of methods that can be used to improve BCI recognition and sense of control. The methods described above are simple and easy to implement, but it is important to apply them in the right way. Detection accuracy is important, but often there are trade-offs to consider. The application provides another angle to improve the interaction, as its design affects the cost of errors, functional requirements on the interaction, and the user expectations.

Future research should investigate the effectiveness of the methods in their context of use, as well as the optimal settings for these methods to obtain the right balance between functionality and meeting the user's expectations for maximum acceptance.

7. ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of the BrainGain Smart Mix Programme of the Netherlands Ministry of Economic Affairs, the Netherlands Ministry of Education, Culture and Science.

[5]Bitrate is also referred to as information transfer rate. For BCIs this is generally expressed in bits per minute.

[6]However, as Steve Jobs warned us: "People don't know what they want, until you show it to them" [21].

8. REFERENCES

[1] G. Abowd, A. Dey, P. Brown, N. Davies, M. Smith, and P. Steggles. Towards a better understanding of context and context-awareness. In *Handheld and Ubiquitous Computing*, pages 304–307. Springer, 1999.

[2] M. Baldauf, S. Dustdar, and F. Rosenberg. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4):263–277, 2007.

[3] A. Bashashati, M. Fatourechi, R. Ward, and G. Birch. A survey of signal processing algorithms in brain–computer interfaces based on electrical brain signals. *Journal of Neural engineering*, 4:R32, 2007.

[4] B. Blankertz, M. Kawanabe, R. Tomioka, F. Hohlefeld, V. Nikulin, and K. Müller. Invariant common spatial patterns: Alleviating nonstationarities in brain-computer interfacing. *Advances in Neural Information Processing Systems*, 20, 2008.

[5] L. Deng and X. Huang. Challenges in adopting speech recognition. *Commun. ACM*, 47(1):69–75, Jan. 2004.

[6] G. Fabiani, D. McFarland, J. Wolpaw, and G. Pfurtscheller. Conversion of EEG activity into cursor movement by a brain-computer interface (BCI). *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 12(3):331–338, 2004.

[7] E. Felton, N. Lewis, S. Wills, R. Radwin, and J. Williams. Neural signal based control of the Dasher writing system. In *Neural Engineering*, pages 366–370. IEEE, 2007.

[8] S. Goderis. *On the separation of user interface concerns: A Programmer's Perspective on the Modularisation of User Interface Code*. PhD thesis, 2008.

[9] H. Gürkök, G. Hakvoort, and M. Poel. Evaluating user experience with respect to user expectations in brain-computer interface games. In *Proceedings of the 5th International Brain-Computer Interface Conference, BCI 2011, Graz, Austria*, pages 348–351. Verlag der Technischen Universität Graz, 2011.

[10] H. Gürkök and A. Nijholt. Brain-computer interfaces for multimodal interaction: A survey and principles. *International Journal of Human-Computer Interaction*, 28(5):292–307, 2012.

[11] C. Holt. Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, 20(1):5–10, 2004.

[12] Y. Huang and C. Suen. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(1):90–94, 1995.

[13] R. Jacob. Eye movement-based human-computer interaction techniques: Toward non-command interfaces. *Advances in human-computer interaction*, 4:151–190, 1993.

[14] R. Jacob and K. Karn. Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. *Mind*, 2(3):4, 2003.

[15] S. Kleih, F. Nijboer, S. Halder, and A. Kübler. Motivation modulates the P300 amplitude during brain-computer interface use. *Clinical Neurophysiology*, 121(7):1023–1031, 2010.

[16] J. Kronegg, S. Voloshynovskiy, and T. Pun. Analysis of bit-rate definitions for brain-computer interfaces. In *HCI'05*, volume 12, page 18, 2005.

[17] L. Lam and S. Suen. Application of majority voting to pattern recognition: An analysis of its behavior and performance. *Systems, Man and Cybernetics, Part A, IEEE Trans.*, 27(5):553–568, 1997.

[18] S. Mason and G. Birch. A brain-controlled switch for asynchronous control applications. *Biomedical Engineering, IEEE Trans.*, 47(10):1297–1307, 2000.

[19] S. Mason and G. Birch. A general framework for brain-computer interface design. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 11(1):70–85, 2003.

[20] C. Mühl, H. Gürkök, D. Plass-Oude Bos, M. Thurlings, L. Scherffig, M. Duvinage, A. Elbakyan, S. Kang, M. Poel, and D. Heylen. Bacteria hunt. *Journal on Multimodal User Interfaces*, 4(1):11–25, 2010.

[21] S. Murugesan. What we can learn from steve jobs. *IT Professional*, 13(6):6–8, 2011.

[22] L. Nigay and J. Coutaz. A design space for multimodal systems: concurrent processing and data fusion. In *Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*, pages 172–178, New York, NY, USA, 1993. ACM.

[23] R. Nuray and F. Can. Automatic ranking of information retrieval systems using data fusion. *Information Processing and Management*, 42(3):595 – 614, 2006.

[24] S. Oviatt. Multimodal interfaces. In *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*, pages 413–428. Lawrance Erlbaum Associates, New York, NY, USA, 2nd edition, 2008.

[25] R. Panicker, S. Puthusserypady, and Y. Sun. An asynchronous P300 BCI with SSVEP-based control state detection. *Biomedical Engineering, IEEE Transactions on*, 58(6):1781 –1788, 2011.

[26] S. Perdikis, H. Bayati, R. Leeb, and J. d. R. Millán. Evidence accumulation in asynchronous BCI. *IJ of Bioelectromagnetism*, 13(3):131–132, 2011.

[27] G. Pfurtscheller, B. Z. Allison, C. Brunner, G. Bauernfeind, T. Solis-Escalante, R. Scherer, T. O. Zander, G. Mueller-Putz, C. Neuper, and N. Birbaumer. The hybrid BCI. *Frontiers in Neuroprosthetics*, 4:30, 2010.

[28] G. Pfurtscheller, G. Müller, J. Pfurtscheller, H. Gerner, and R. Rupp. Thought-control of functional electrical stimulation to restore hand grasp in a patient with tetraplegia. *Neuroscience letters*, 351(1):33–36, 2003.

[29] D. Plass-Oude Bos, M. Poel, and A. Nijholt. A study in user-centered design and evaluation of mental tasks for BCI. *Advances in Multimedia Modeling*, pages 122–134, 2011.

[30] D. Plass-Oude Bos, B. Reuderink, B. L. A. van de Laar, H. Gürkök, C. Mühl, M. Poel, D. K. J. Heylen, and A. Nijholt. Human-computer interaction for BCI games: Usability and user experience. In *CYBERWORLDS*, pages 277–281. IEEE, 2010.

[31] B. Reuderink. *Robust brain-computer interfaces*. PhD thesis, Enschede, the Netherlands, October 2011.

[32] S. Roberts and W. Penny. Real-time brain-computer interfacing: A preliminary study using bayesian learning. *Medical and Biological Engineering and computing*, 38(1):56–61, 2000.

[33] A. Royer and B. He. Goal selection versus process control in a brain-computer interface based on sensorimotor rhythms. *Journal of neural engineering*, 6:016005, 2009.

[34] R. Scherer, M. Proll, B. Allison, and G. Muller-Putz. New input modalities for modern game design and virtual embodiment. In *Virtual Reality Workshops (VR), 2012 IEEE*, pages 163 –164, March 2012.

[35] C. Shannon. The redundancy of english. In *Cybernetics; Transactions of the 7th Conference, New York: Josiah Macy, Jr. Foundation*, pages 248–272. Diaphanes Verlag Berlin/Zurich, 1950. Reprinted 2003.

[36] G. Townsend, B. Graimann, and G. Pfurtscheller. Continuous EEG classification during motor imagery-simulation of an asynchronous BCI. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 12(2):258–265, 2004.

[37] M. Turunen, J. Hakulinen, A. Melto, T. Heimonen, T. Laivo, and J. Hella. SUXES – user experience evaluation method for spoken and multimodal interaction. In *INTERSPEECH*, 2009.

[38] B. van de Laar, H. Gürkök, D. Plass-Oude Bos, M. Poel, and A. Nijholt. BCI as an additional control in an immersive game; a within-subjects comparison. 2012. In submission.

[39] B. van de Laar, D. Plass-Oude Bos, B. Reuderink, M. Poel, and A. Nijholt. How much control is enough? optimizing fun with unreliable input, 2011. CTIT technical report series, TR-CTIT-11-25.

[40] M. van Gerven, J. Farquhar, R. Schaefer, R. Vlek, J. Geuze, A. Nijholt, N. Ramsey, P. Haselager, L. Vuurpijl, S. Gielen, and P. Desain. The brain–computer interface cycle. *Journal of Neural Engineering*, 6(4), 2009.

[41] J. Williamson, R. Murray-Smith, B. Blankertz, M. Krauledat, and K. Müller. Designing for uncertain, asymmetric control: interaction design for brain–computer interfaces. *International Journal of Human-Computer Studies*, 67(10):827–841, 2009.

[42] J. Wolpaw. Brain–computer interfaces as new brain output pathways. *The Journal of Physiology*, 579(3):613–619, 2007.