Chapter 3

Speeding up Hands-free Text Entry

T. Felzer and R. Nordmann

3.1 Introduction: Why is Thinking about Hands-free Text Entry Important, Anyway?

Computers are everywhere nowadays: they can, for example, be used for reading or composing emails, surfing the Internet or managing incoming and outgoing phone calls. They can also realize specialized applications, such as controlling an electric wheelchair (Felzer and Freisleben, 2002*a*) or monitoring the heartbeat of a patient (and placing a rescue phone call in case of an emergency).

The ability to use a computer is therefore of huge importance, specifically for persons with disabilities, where the computer is sometimes the only portal to the outside world. Unfortunately, the standard way of operating a computer, *i.e.*, with the help of a keyboard and a manual mouse, is unsuitable for someone who cannot reliably use hands or arms. Thus, developing hands-free alternatives, especially for text entry, is not only a very promising task, but simply a plain necessity in our modern society.

This chapter talks about such an alternative interface, the HaMCoS software tool, which enables its user to drive the computer mouse without using the hands. As a supplement of the HaMCoS software, the *Mouse Editor* is described which aims at text entry with the help of an on-screen keyboard. One main focus is the possibility of speeding up the entry rate by using the word completion feature of the *Mouse Editor*. Several experiments comparing text entry with and without word completion were conducted, and the results are discussed.

The remainder of this chapter is organized as follows. The next section gives some general remarks on hands-free text entry. In particular, it deals with related work on alternative input interfaces and with word completion as a means to reduce the physical load on the user of such an interface. After that, the interface solution to be evaluated here will be looked at in detail: firstly, the input method and the practical realization thereof are briefly explained; secondly, the application allowing the user to enter text with the help of the aforementioned input method is described; and finally, the results received with this setting are reported.

The last two sections conclude the chapter by presenting a short summary and the list of references.

3.2 Some General Remarks on Hands-free Text Entry

3.2.1 Related Work: How to Operate a PC Without Using the Hands

A large number of alternative input devices, assisting persons with disabilities in gaining access to personal computers, have been introduced in the past. Since hands and arms are often inappropriate in this respect as the basic communication channel, various bio-signals have been examined regarding their applicability to control purposes.

The idea behind those *bio-signal interfaces* is to monitor the time series of a suitable bodily function of the user, process the data by recognizing recurring patterns, and generate control commands accordingly. The user can therefore decide which commands are executed by intentionally altering the monitored bio-signal and issuing the desired patterns (Felzer, 2002).

One example for a bio-signal employed in human-computer interaction is given by the electro-oculography (EOG) signal, which makes use of the dipole characteristics of the eyes (Lileg *et al.*, 1999). Electrodes placed next to the eyes tell EOG-based interfaces the angle of the eyes relative to the head. This information is used to find out where the user is looking. The interface can move a cursor on the computer screen corresponding to the user's gaze across, for example, a virtual (or *on-screen*) keyboard. The user is thus able to select characters simply by looking at them. The system might then accept that character, if the user's gaze remained over the same location for a certain amount of time (often referred to as the *dwell latency*). As a consequence, the user can 'type' short messages.

In some eye-typing systems (for an overview, see Majaranta and Räihä, 2002), information on the line-of-sight is combined, *e.g.,* with an intentional blink or a manual switch. In addition, a number of gaze-related interfaces use cameras and *computer vision* (*e.g.,* Matsumoto *et al.*, 2001) for determining the user's gaze direction, which often leads to a much more expensive system (as compared to an EOG-based interface).

Anyway, all eye-tracking systems share one fundamental problem: they occupy the eyes. The user is *not* free to look wherever he or she wants. Besides, there is the so-called 'Midas Touch problem', which means that the user staring at the screen without altering the gaze direction (*e.g.*, to read something) might result in erroneous selections.

The bio-signal examined in so-called *brain-computer interfaces* (BCI's, see Felzer and Freisleben, 2002*c*, or, for an overview, Wolpaw *et al.*, 2002) are the *brain-waves* of the user. A BCI tries to analyze some representation of the on-going electroencephalography (EEG) signal, in order to find certain elements produced at will. This analysis often involves classification with the help of an artificial neural net (ANN, *e.g.,* Felzer and Freisleben, 2003).

There is, however, a large variety of types of elements that are searched for in the EEG signal. For example, the BCI of Birbaumer (see Hinterberger *et al.,* 2003) is based on the detection of *slow cortical potentials* (SCP's), which are voltage shifts in the brain-electrical activity that can be produced wilfully (after extensive training).

The *adaptive brain interfaces* of del R. Millán (2003) depend on different mental tasks (*e.g.*, performing non-trivial arithmetic operations or mentally uttering language) and on how distinct the corresponding EEG recordings are.

McFarland *et al.* (2005) observe the *mu or beta rhythm* amplitude (brain-electrical activity at certain frequencies), which can be controlled at will after some neuro-feedback training.

There are many additional examples that may also be listed here, but they all suffer from (at least) two common drawbacks. First, they are awfully slow: a typical paper in this respect (Neuper *et al.*, 2003) talks about a 'typing' rate of approximately one character per minute. Second, BCI systems are extremely sensitive to noise, since the EEG consists of very tiny potentials that have to be amplified by a factor on the order of 10000, so the final EEG recordings are contaminated by all sorts of artefacts related to muscular activity affecting the so-called electromyography (EMG) signal (Goncharova *et al.*, 2003).

On the other hand, the big advantage of EEG-based systems is that they do not require any physical activity at all. One way to benefit from the positive effects of EEG, without suffering too much from its problems is to use a hybrid system, such as the *Cyberlink* interface (*e.g.*, Doherty *et al.,* 2000), which combines EOG, EMG, and EEG signals.

Barreto *et al.* (1999) rely on a combination of EEG and EMG signals to enable the user to move the mouse cursor on a computer screen (although in their system, the EEG merely switches EMG control on or off).

Finally, a growing number of interface systems completely renounce the error-prone EEG processing. Examples include the approach of Moon *et al.* (2003), which detects the face direction with the help of a camera and EMG signals originating from certain neck muscles, or the system introduced by Tarng *et al.* (1997), which exclusively concentrates on the EMG for driving a mouse cursor in two dimensions on a computer screen.

The interface that is applied here as a basis for the results being reported in the next section relies solely on muscular activity, too. However, instead of analyzing signals picked up by conventional, *passive* EMG electrodes (which are subject to external electromagnetic interference), it is based on a Piezo element actively generating voltage potentials according to the paradigm of *electromechanical coupling* and corresponding to the contraction of a single muscle of choice. As a consequence, the system on the one hand requires only slightly more physical effort than a BCI (*e.g.,* it can be operated by merely raising the eyebrow), but is, on the other hand, much more robust and much less sensitive to noise than an EEG-based BCI or an interface system relying on *conventional* EMG signals.

3.2.2 Employing Word Completion to Reduce the Physical Load on the User

Assistive devices often employ word completion (or word *prediction*) as a means to reduce the physical load on the user. Such a routine tries to guess the word the user is about to type on the basis of the initial characters typed by the user. The word completion routine then offers a list of completion candidates, and the user can save keystrokes by selecting such a candidate. This, however, does not necessarily mean a speedup of the typing rate (although it does here, as will be demonstrated later), since there is a trade-off in that having to scan through the candidate list also increases the cognitive load on the user (see Koester and Levine, 1994).

The completion candidates are mostly determined using statistical methods (see Stocky *et al.*, 2004). The most important strategies in this respect refer to *frequency* (of words in a given context/language) or *recency* (which leads to the suggestion of candidates most recently typed by the user). There are also approaches taking syntactical information of the text typed so far into account (*e.g.*, Garay-Vitoria and González-Abascal, 1997), but the syntax is not considered by the realization in the next section.

3.3 Hands-free Operation Based on Intentional Muscle Contractions

3.3.1 Controlling the Mouse Using the HaMCoS Software

In this chapter, the use of an input method relying on intentional muscle contractions (Felzer and Freisleben, 2002*b*) is reported. The method is applied to text entry in order to investigate the potential benefits of word completion for assistive technology. The muscle-based input method is implemented in a software system called HaMCoS, which stands for *Ha*nds-free *M*ouse *Co*ntrol *S*ystem.

HaMCoS interfaces with the standard microphone input and was developed under the operating system Windows® XP from Microsoft®. The system tries to detect muscle contractions by simply inspecting the amplitude of an ‘EMG-like’ signal – picked up by some acquisition device – relating to a single muscle of choice of the user (which he or she can reliably control). Instead of a conventional (*passive*) EMG electrode, a Piezo element is used, which actively generates voltage potentials according to the paradigm of electromechanical coupling. Therefore, the system – unlike those based on ‘normal’ EMG – is selectively responding to contractions of the single dedicated muscle.

For example, if the user is able to wilfully contract the brow muscle, the device could be attached to his/her forehead with the help of a headband. All the software has to do is detect intentional muscle contractions – it translates the input into a sequence of single (CE1) or double (CE2) *contraction events*. In the example, a CE1 corresponds to the user frowning once, and a CE2 to frowning twice, in quick succession.

Processing the contraction sequence on the basis of the transition diagram shown in Figure 3.1 leads to an internal state, which denotes the direction of mouse cursor movement to be initiated by the HaMCoS software. At the beginning, the program is in the 'STOP' state, and the mouse doesn't move. After a CE2, the mouse starts to continually move left, until another contraction is detected. A CE1 then changes movement direction (*e.g.*, from 'LEFT' to 'UP'), and when the mouse cursor has finally reached the target position, it can be stopped using a CE2.

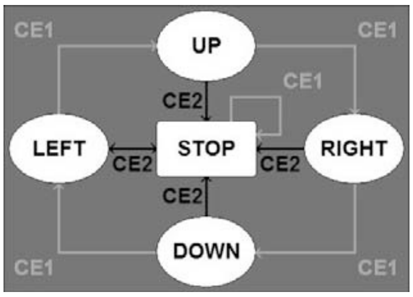

Figure 3.1. Transitions among five states governing mouse motion

Whenever the mouse cursor stops (*i.e.,* the program state reverts from a moving to the 'STOP' state), the software issues a mouse click, whose type (left or right, single or double, drag start, *etc*.) may be selected by the user (Felzer *et al.*, 2005).

In such a way, a user of the HaMCoS software can fully control a PC provided it exclusively relies on mouse input (see also Felzer and Nordmann, 2005). However, this latter condition requires some specialized software in addition to the *Main Module* of HaMCoS – which merely emulates mouse actions – *e.g.*, an on-screen keyboard for text entry.

3.3.2 Editing Text Files With the *Mouse Editor*

Although the user is free in choosing any software he or she likes, HaMCoS comes with a comprehensive framework containing auxiliary applications for every purpose which are optimized for use in combination with the *Main Module:* for example, framework applications do not distinguish between different click types (*e.g.,* left or right), so there is no need for the obligatory selection step when reverting to 'STOP'.

One of the foremost challenges as far as 'hands-free' human-computer communication is concerned is related to text entry or keyboard commands, in fact everything that – in the standard case – requires the use of a keyboard. HaMCoS offers the possibility to completely control the computer mouse by intentionally contracting a single muscle of choice. As a consequence, keyboard input can be controlled equally easy, if it is somehow 'mapped' to mouse input by the HaMCoS framework. The tool designated to be used as a means to enter arbitrary text is the *Mouse Editor* depicted in Figure 3.2.

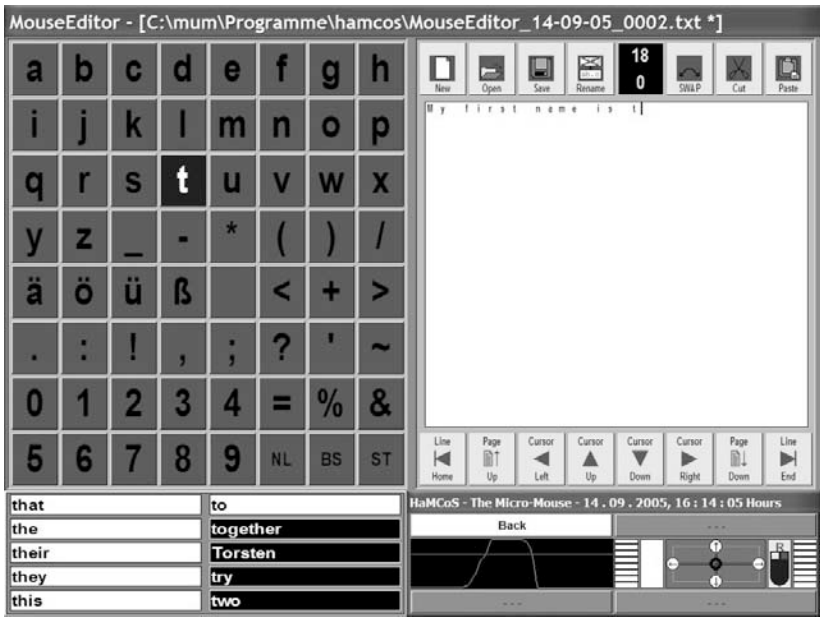

Figure 3.2. Using the *Mouse Editor* to enter text

The *Mouse Editor* is responsible for the creation and the modification of text files. It offers a functionality comparable to the operating system's *Notepad* editor. The difference is, however, that the *Mouse Editor* does not require any manual keyboard input.

The editor window is divided into two parts, showing a virtual keyboard on the left and the text file being edited containing a flashing cursor – as well as a menu row above and one row of navigation buttons below – on the right.

The virtual keyboard consists of 64 buttons, arranged in a square grid with eight rows and eight columns, each one associated with a certain keyboard function. In most cases, clicking one of those buttons corresponds to inserting the appropriate character at the current cursor position – the exceptions being the buttons in row number eight labelled 'NL', 'BS', and 'ST':

- the 'NL' button corresponds to a *newline* character, *i.e.*, the cursor moves to the first position of a newly inserted line;
- the 'BS' button does the same as the *backspace* key of a regular keyboard, *i.e.*, it deletes the character to the left of the cursor or the last *newline* character if the cursor is at the beginning of a line;
- the 'ST' button serves as a *shift* key: when this button is clicked once, the keyboard immediately shows a shifted version of the available characters (in most cases either capitalized or unchanged) and reverts back to the un-shifted view after a click on any other button.

The main portion on the right part of the editor screen consists of a sliding window showing 24 lines and 32 columns of the text file being edited, as well as a flashing cursor, which marks the current position. Although the window's size is quite modest, the files can be of arbitrary dimensions, and the window is automatically scrolled in such a way that the cursor always remains visible.

The navigation buttons may be used to alter the current position (*e.g.*, by sending the cursor to the end of the current line) without changing the text. In combination with shifted mode, these buttons are used for marking portions of text. Marked text can be copied to the Windows® Clipboard (by clicking on the 'Cut' button of the menu row when in shifted mode) or it can be replaced by inserted text. This refers to either the current Clipboard contents (which are inserted each time the top right 'Paste' button is clicked in shifted mode) or a single character entered with the help of the virtual keyboard. The four leftmost buttons of the menu row are responsible for creating, loading, saving, and renaming text files.

In addition to this '*Notepad* functionality', the *Mouse Editor* offers word completion: after entering the initial characters of a word (*i.e.*, the *prefix*) the user is presented with a list of candidates as to what word he or she is about to 'type'. Clicking one of the candidate buttons replaces the prefix with the completed word.

The candidate list actually contains two kinds of words: first, a 'general' list with the 10000 most frequent words in the English language (according to 'http://wortschatz.uni-leipzig.de') is searched for suitable completions, and second, the same is done with a 'personal' list containing the words most frequently 'typed' by the user. The resulting candidate lists are sorted alphabetically, in order to make it easier for the user to decide whether the intended word is displayed.

Each time a text file is saved with the help of the *Mouse Editor*, the file to be written to disk is analyzed and the 'personal' list is updated. That way, the word completion routine adapts to the user's own writing style and is even able to suggest frequently used technical terms or proper nouns as completion candidates.

Finally, it should be mentioned that the *Mouse Editor* is primarily intended to be operated with the help of mouse actions alone, *independent* from any input method. In other words, although the *Mouse Editor* is part of the HaMCoS framework, it might as well be run using a conventional (*manual*) mouse device.

3.3.3 Experimental Results With and Without Word Completion

In order to find out whether the word completion feature of the *Mouse Editor* speeds up text entry (in addition to the 'keystroke saving'), several experiments with one male, 34-year-old subject have been conducted. The subject was diagnosed with Friedreich's Ataxia in 1985 – he is obliged to use a wheelchair and also has problems related to hand/arm coordination. However, he is still able to operate keyboard and mouse (which makes the results relating to different input methods somewhat comparable). Besides, he practised working with HaMCoS for several weeks (on average, less than 30 minutes per day).

The subject's task was to copy the first two paragraphs of the preface of volume 1 of Rumelhart and McClelland's (1986) book as quickly and accurately as possible using the 'normal' keyboard (in a standard editor), a manual mouse, and the HaMCoS sensor. Mouse and HaMCoS trials were repeated three times each with word completion *not* used *at all* in the first round and with an empty 'personal' list in the second round.

For the third round, the copied text was saved (and therefore analyzed) by the *Mouse Editor*, so the word completion was then optimized for this now *known* text fragment – although the conditions in this round were admittedly rather idealized, it can certainly provide an upper bound for using word completion *with* 'personal' list. The results are presented in table 3.1.

Table 3.1. Times needed to copy two paragraphs containing 15 lines, 164 words, and 998 'characters' (including *newline* characters). Note that mistakes (wrong characters *etc*.) were corrected immediately in each trial, so the results report the overall values. The trials conducted using keyboard and manual mouse comprised all 15 lines each, while the HaMCoS trials were subdivided into five 3-line packets containing 204, 185, 206, 192, and 211 characters, respectively.

Input Method	Degree of Completion	Time Needed (minutes)
Keyboard	No Completion	32'
Manual Mouse	No Completion	79'
	General Completion	59'
	Optimized Completion	43'
HaMCoS	No Completion	308' (65' 57' 63' 61' 62')
	General Completion	228' (51' 41' 47' 43' 46')
	Optimized Completion	164' (34' 30' 35' 33' 32')

The table indicates that 'general completion' leads to a speedup of about 25% and 'optimized completion' to a speedup of about 45% compared to the 'no completion' round for both manual mouse and HaMCoS sensor.

When comparing the 'no completion' round with the manual mouse and the figures for the (very slow) HaMCoS input method, it can be seen that optimized completion cannot fully compensate for the overhead caused by not being able to control the hands, but it can at least reduce that surcharge from a factor of about four ($4 \times 79 \cong 308$) to a factor of about two ($2 \times 79 \cong 164$).

3.4 Chapter Conclusion

An alternative human-computer interface representing a means of using a computer for entering text which does *not* require the user's hands or arms has been described and evaluated. The input method is based on intentional muscle contractions and requires extremely little physical effort.

As a compensation for the relatively slow input speed possible with the interface, the text entry is combined with a word completion feature which aims at guessing the word the user is about to type. The word completion routine looks for completion candidates in two lists: a 'general' list containing the most frequent words in the English language and a 'personal' list with the words chosen most often by the user. One of the consequences of the mechanism which constantly updates the 'personal' list is that the word completion routine adapts to the user's own writing style.

As can be seen from the results, the word completion feature indeed has the potential to reduce the time required to copy a moderately long text and thus to speed up text entry. In other words, the temporal gain is not eaten up by the increased cognitive load on the user (caused by having to scan through the list of completion candidates).

Since a person who cannot control his/her hands – using HaMCoS and the *Mouse Editor* with *optimized* word completion – is able to enter text in only about double the time compared to someone using a manual mouse (*without* word completion), the system in its ultimate form represents a true alternative for persons with severe physical disabilities. Additional experiments with a larger number of potential users of the system are to be conducted in the near future.

3.5 References

Barreto AB, Scargle SD, Adjouadi M (1999) A real-time assistive computer interface for users with motor disabilities. ACM SIGCAPH Computers and the Physically Handicapped 64: 6-16

del R Millán J (2003) Adaptive brain interfaces. Communications of the ACM 46(3): 74-80

Doherty E, Stephenson G, Engel W (2000) Using a cyberlink mental interface for relaxation and controlling a robot. ACM SIGCAPH Computers and the Physically Handicapped 68: 4-9

Felzer T (2002) Verwendung verschiedener Biosignale zur Bedienung computergesteuerter Systeme (Using various kinds of bio-signals for controlling computer-mediated systems). Wissenschaftlicher Verlag Berlin, Ph.D. Thesis (German), ISBN 3-932089-99-5

Felzer T, Fischer R, Groensfelder T, Nordmann R (2005) Alternative control system for operating a PC using intentional muscle contractions only. In: Online-Proceedings of the 2005 CSUN Conference on Technology and Persons with Disabilities, Los Angeles, California, USA. Available at:
http:// www.csun.edu/cod/conf/2005/proceedings/2121.htm

Felzer T, Freisleben B (2002*a*) HaWCoS: The 'hands-free' wheelchair control system. In: Proceedings of the Fifth International ACM SIGCAPH Conference on Assistive Technologies, ACM Press, Edinburgh, UK, pp 127-134

Felzer T, Freisleben B (2002*b*) Controlling a computer using signals originating from muscle contractions. In: METMBS '02 - Proceedings of the 2002 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences, Vol 2, CSREA Press, Las Vegas, Nevada, USA, pp 336-342

Felzer T, Freisleben B (2002*c*) BRAINLINK: A software tool supporting the development of an EEG-based brain-computer interface. In: METMBS '02 - Proceedings of the 2002 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences, Vol 2, CSREA Press, Las Vegas, Nevada, USA, pp 329-335

Felzer T, Freisleben B (2003) Analyzing EEG signals using the probability estimating guarded neural classifier. IEEE Transactions on Neural Systems and Rehabilitation Engineering 11(4): 361-371

Felzer T, Nordmann R (2005) How to operate a PC without using the hands. In: Proceedings of the Seventh International ACM SIGCAPH Conference on Computers and Accessibility, ACM Press, Baltimore, MD, USA, pp 198-199

Garay-Vitoria N, González-Abascal J (1997) Intelligent word-prediction to enhance text input rate (a syntactic analysis-based word-prediction aid for people with severe motor and speech disability). Proceedings of the 2nd International Conference on Intelligent User Interfaces, Orlando, Florida, USA, pp 241-244

Goncharova II, McFarland DJ, Vaughan TM, Wolpaw JR (2003) EMG contamination of EEG: Spectral and topographical characteristics. Clinical Neurophysiology 114: 1580-1593

Hinterberger T, Kübler A, Kaiser J, Neumann N, Birbaumer N (2003) A brain-computer interface (BCI) for the locked-in: comparison of different EEG classifications for the thought translation device. Clinical Neurophysiology 114: 416-425

Koester HH, Levine SP (1994) Modeling the speed of text entry with a word prediction interface. IEEE Transactions on Rehabilitation Engineering 2(3):177-187

Lileg E, Wiesspeiner G, Hutten H (1999) Evaluation of the EOG for communication through eye movements. In: Proceedings of the 10th European Conference on Eye Movements, Utrecht, The Netherlands

Majaranta P, Räihä KJ (2002) Twenty years of eye typing: systems and design issues. In: Proceedings of the Symposium on Eye Tracking Research & Applications, New Orleans, Louisiana, USA, pp 15-22

Matsumoto Y, Ino T, Ogsawara T (2001) Development of intelligent wheelchair system with face and gaze based interface. In: Proceedings of the 10th IEEE International Workshop on Robot-Human Interactive Communication (Bordeaux and Paris, France), pp 262-267

McFarland DJ, Sarnacki WA, Vaughan TM, Wolpaw JR (2005) Brain-computer interface (BCI) operation: Signal and noise during early training sessions. Clinical Neurophysiology 116: 56-62

Moon I, Kim K, Ryu J, Mun M (2003) Face direction-based human-computer interface using image observation and EMG signal for the disabled. In: Proceedings of the IEEE International Conference on Robotics and Automation, Vol 1, Taipei, Taiwan, IEEE Press, pp 1515-1520

Neuper C, Müller GR, Kübler A, Birbaumer N, Pfurtscheller G (2003) Clinical application of an EEG-based brain-computer interface: A case study in a patient with severe motor impairment. Clinical Neurophysiology 114: 399-409

Rumelhart DE, McClelland JL (ed.) (1986) Parallel distributed processing: explorations in the microstructures of cognition. MIT Press, Cambridge, MA, USA

Stocky T, Faaborg A, Liebermann H (2004) A commonsense approach to predictive text entry. In: Proceedings of the Conference on Human Factors in Computing Systems (CHI 2004), ACM Press, Vienna, Austria, pp 1163-1166

Tarng YH, Chang GC, Lai JS, Kuo TS (1997) Design of the human/computer interface for human with disability using myoelectric signal control. In: Proceedings of the 19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Vol 5, IEEE Press, Chicago, Illinois, USA, pp 1909-1910

Wolpaw JR, Birbaumer N, McFarland DJ, Pfurtscheller G, Vaughan TM (2002) Brain-computer interfaces for communication and control. Clinical Neurophysiology 113: 767-791