

An Input Method for Human–Computer Interaction based on Muscle Control

T. Felzer and B. Freisleben

Abstract—An input method for human-computer interaction is presented that allows one to communicate with a computer by contracting any muscle of choice, e.g. a facial muscle. This makes it possible to perform all sorts of computer-controllable actions just by raising the eyebrow, i.e. almost effortlessly. Possible applications include composing e-mail messages, navigating in a web-browser or using various software applications. Since the system is robust and almost insensitive to any kind of noise, it could also be used to control moving objects, such as a mobile robot or an electrical wheelchair. The system can therefore be of great help for persons with disabilities, but it may of course be used by able-bodied persons as well, e.g. for controlling home entertainment devices like TV sets or CD players. The potential benefit of the system is demonstrated by means of a spelling device as an example application. It will be shown that it is possible to use the system as a substitution for the standard computer keyboard, allowing to “type” at a rate of at least 4 to 5 characters per minute.

Index Terms— Human-computer interaction, EEG analysis, muscle control.

I. INTRODUCTION

COMMUNICATING with a computer normally means using a keyboard and a mouse. However, this *standard* way is inappropriate for persons suffering from severe physical disabilities, because it obviously requires the reliable use of the hands. Some neurological diseases, e.g. amyotrophic lateral sclerosis (ALS), can end in a so-called “locked-in” state, a situation where a *patient’s mobile mind is locked in an immobile body*. In addition to not being able to use their hands, those patients are unable to speak, so any kind of voice recognition system cannot improve their situation either.

In this paper, an input method is described which requires its user to have reliable control over a single muscle only. If a patient can produce a simple muscle tension at will, e.g. by clenching the teeth, raising the eyebrow or blinking with the eye, he or she is able to use the system underlying the method introduced here. The idea behind the system is to search the EEG or EMG signal of a subject for wilfully generated muscle contractions, thereby transforming the signal into a stream of discrete contraction and non-contraction phases, and finally producing appropriate output commands – controlling a *target application* based on the respective order of two types of *contraction events*.

The feasibility of the approach is demonstrated with the help of a *spelling device* as an example target application. Since the concrete realization allows “typing” at a rate of about 4–5

characters per minute in a stable and robust way, the system represents an effective alternative for physically disabled people – who now have the chance to write e.g. e-mails with extremely little physical effort. Moreover, operating the system does not depend on particular conditions – neither concerning the recording environment nor concerning the physical condition of the user (as long as he or she is able to control a single muscle of choice). Therefore, the system can also be an assistive device for able-bodied people – with an ample number of possible applications, such as an “eyebrow-operated” emulation of an infrared remote control for the home entertainment center or a phone headset allowing to dial by “frowning”.

The paper is organized as follows. Section II contains a discussion of related work and is designed to answer the question why the introduced system is like it is. This is followed by a description of the architecture of the system (applied to the example of a spelling device) as well as of the actual software implementation. Section V comments on first results obtained with the example device, which document its overall performance. The paper concludes with a short summary and some thoughts on related applications and future work in section VI.

II. RELATED WORK

Various types of alternative input methods, which somehow deviate from the standard keyboard and mouse approach, have been reported in the literature. They all are based on different kinds of input signals, some particularly addressing persons with disabilities, others not limited to a certain group of people.

A common principle often used by an alternative input system may be denoted by the term *scanning device*. In this context, a number of options are cyclically highlighted, and the user of the system can initiate the execution of the selected option by generating a particular trigger signal at the right time. An example of such a system is given by the wheelchair control device presented in [1]. In this case, there are five options to choose from (standing for the movement directions of the electrical wheelchair to be controlled), which are symbolized by five LED’s cyclically lit. When the LED belonging to the desired direction is lit (i.e. “highlighted”), the user can set the wheelchair in motion (into the corresponding direction) by operating a special headswitch. Pressing the headswitch again halts the wheelchair and restarts the scanning process.

An example based on a totally different principle is the approach of Birbaumer et al. [2] which records and analyzes the EEG of a subject. That device tries to detect slow cortical potentials¹, and since humans can learn to regulate their SCP’s

T. Felzer is with the Department of Computer Science, Darmstadt University of Technology, Alexanderstr. 10, D-64283 Darmstadt, Germany

B. Freisleben is with the Department of Mathematics and Computer Science, University of Marburg, Hans-Meerwein-Str., D-35032 Marburg, Germany.

¹Slow cortical potentials (SCP’s) represent a certain kind of polarizations in the EEG.

voluntarily after prolonged bio-feedback training, the subject can intentionally select characters by varying his or her brain-waves (see also [3], [4]). The system is very sensitive to noise, though, and a lot of care has to be taken in order to cope with that.

Although the (disabled) user surely is very grateful for the development of the device (since self-initiated control of the environment might be totally impossible or at least extremely difficult without), using the device is fatiguing, troublesome, and awfully slow. Therefore, someone who does not depend on the particular support offered by the device will definitely not use it in practice.

Birbaumer's work (see also [5]) actually represents an example of an EEG-based brain-computer interface (BCI). In general, a BCI is a system inspecting the brain-electrical activity (the EEG) of a subject and generating output commands based on patterns recognized by a computer. Since this idea has first been mentioned (in 1973 [6]), many groups of researchers have started to work on the development of a BCI, and because this topic is currently very popular, the number of such groups still seems to grow every year [7], [8], [9], [10], [11], [12], [13], [14].

For instance, the group around Wolpaw and McFarland [15], [16] developed a BCI based on the 8–12 Hz mu rhythm activity in the EEG. They made use of the fact that humans are able to (learn to) control their mu rhythm. The amplitudes of the mu rhythm recorded from subjects' skulls were translated into one- or two-dimensional cursor movement (on a video screen) – with an accuracy of at most 70% in the two-dimensional case.

Pfurtscheller and his colleagues [17], [18] used neural network classifiers to analyze the subject's EEG activity. The network's task was to find out which of three different movements was performed (or “mentally” planned) by the subject. In order to optimize the classification accuracy, they had to discard EEG segments contaminated by all sorts of muscle contraction artifacts.

Anderson et al. [19] (see also [20]) also use a neural network classifier to determine which one of five different mental tasks a subject is performing. Their idea was to enable the subject to communicate with a computer by composing sequences of mental states – the classification accuracy of 73% or less on untrained segments was rather poor, though.

To summarize, it must be said that EEG-based BCI's only offer *moderate* accuracy levels while having to struggle with a lot of problems related to artifacts caused by muscle contractions and other *noise*, as well as relying on very strict recording conditions.

Examples of systems controlled by eye movements or by EMG signals do *not* suffer from these particular drawbacks.

Degermann et al. [21] describe a system called *Eyegaze* that enables the user to select characters by moving the eyes (which are monitored by a computer with the help of a video camera). The system yields good results and working with it is relatively simple to learn, but its use causes a certain risk of strain. Tecce et al. [22] present a character selection system based on EOG signals (which document the relative position of the eyeballs). The average performance reported there is about one character every 2.6 seconds. Much better results can be obtained using

the eye-controlled device described in [23] (although it requires additional motor control over the eyelid). The system determines the line-of-sight of a subject by examining the reflection of his or her eye illuminated by a near-infrared light source and invokes commands when the subject generates an intentional blink. Since position and orientation of the head are also taken into account, the operation of the device is very stable. However, it requires a lot of special hardware.

Alsayegh [24] describes a system, where the EMG associated with three arm muscles is examined with the goal of distinguishing between twelve different hand gestures, thus allowing gesture-based human-computer interaction. The input method described there is of course *non-standard*, since it does not make use of a keyboard or a mouse – it is, however, inappropriate for helping disabled persons, since it still requires control over the hands.

The interface system introduced in [25] gives a severely disabled person – who can use neither hands nor eyes – a chance to communicate with his environment by contracting the masseter muscle (which is responsible for the closure of the jaws when chewing). The examination of the EMG signal leads to a translation of the contraction durations into dots and dashes. This enables the user of the system to use “Morse code” – having a conversation using this technique takes very long, though.

The approach described in [26] also relies on the analysis of EMG signals – this time targeted at the emulation of a computer mouse. By using a neural network classifier, the system tries to distinguish between five movement patterns, while translating the recognized classes into the displacement of the mouse pointer and the triggering of “mouse clicks”. Although the reported recognition performance is relatively high, the expense of placing no less than five electrodes seems unnecessarily large.

The eye-controlled and EMG based devices surely have to be called stable and useful systems, much less noise-sensitive and allowing better performance rates than BCI's, but controlling a computer is much more direct and can be done (by able-bodied subjects) much faster using the standard way (with keyboard and mouse) than employing that device. Moreover, they require (sometimes *considerable*) motor control, and therefore are unsuitable for some disabled users.

Finally, some thoughts on the commercial system *Cyberlink* (by Brain Actuated Technologies, Inc.) – presented in [27], [28] – shall be added. This very promising system sort of combines EOG, EMG, and EEG signals. The three types of bio-signals are recorded with the help of three electrodes attached to the user's forehead (and kept in place by a headband), amplified (by a factor of 10^5), digitized, and sent to a computer for further processing. There, the input signals are translated into eleven continuous (ten of which correspond to various EEG frequency ranges) and four discretized output signals. The output signals can be used to control any software application running on the analyzing computer.

The system is very compact and universal, and it is therefore often used as the basis for even more sophisticated interfaces (e.g. [29], [30]). However, it is almost as sensitive to noise as any BCI based on EEG signals only. Furthermore, the user of the system has to learn to wilfully alter the brain-waves (which

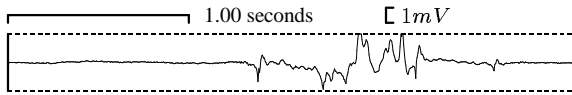


Fig. 1. EEG Trace Contaminated by Muscle Contraction Artifact

may take several months) in order to use them for control purposes. Since there are many applications, where EEG is inappropriate anyway, the statement to use brain-electrical activity often seems to be nothing more than a nice promise.

III. SYSTEM ARCHITECTURE

The system introduced in this paper was designed to exhibit two (in some way *contradicting*) properties. On the one hand, the goal was to develop an input device that requires extremely little physical effort and thus is able to assist people with disabilities. On the other hand, the projected device should be a *practicable* tool that is suitable for “real-world” applications (without relying on “unnatural conditions”) – in other words, it should have the potential of becoming an alternative even for able-bodied people (at least as far as certain specific target applications are concerned).

Evidently, the resulting system had to be some sort of a *compromise*. As can be seen from the BCI examples detailed in the last section, realizing an input device as part of an EEG-based brain-computer interface offers a great deal of advantages, since generating the necessary variations in the brain-waves causes the least possible *physical* effort (since only *mental activities* are involved), but unfortunately, the resulting device would only work – i.e. produce acceptable results – if its user sat perfectly still in front of a computer.

The reason for this is that the EEG signal – which measures tiny potential differences – is extremely sensitive to noise caused by muscle contractions. Fig. 1 shows an EEG trace that documents this sensitivity. The subject this trace originates from does nothing at the beginning of the recording. After about 1.2 seconds, the subject is asked to slightly turn the head. This tiny movement results in a huge burst of activity with amplitudes of more than ten times as high as in the uncontaminated portion (note the rather coarse scale).

These movement-related bursts considerably disturb EEG analysis, since the resulting signal has nothing to do with the user’s thoughts (as the EEG signal *usually* does). Therefore, in order to reliably employ a system based on “true” EEG analysis, its user would either have to be completely *unable* to produce any artifacts, or he or she would have to totally *renounce* any disturbing muscle contraction (which by the way also includes talking) – even the slightest movement of head, eyes or facial muscles, even the smallest change in the environment (another person passing by, a moving elevator several feet away) changes the EEG considerably.

However, the system introduced here was intended to not only work with a (preliminary) spelling device as target application, but also with “real-world” applications – e.g. a control system for an electrical wheelchair – and it soon became clear that the BCI idea was not suitable for that. Not only is the EEG extremely sensitive to noise, there is also a timing problem: a

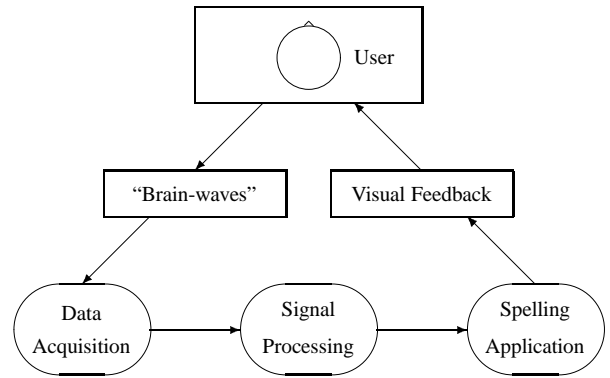


Fig. 2. Basic Architecture

wheelchair control system requires processing the input data in real-time, since the system has to react fast and promptly to the user’s inputs², and it is more than *questionable*, if an EEG-based BCI is able to provide reliable results *and* maintain the kind of reaction time needed in this context.

Therefore, an EEG-based BCI did *not really* seem to be the perfect solution. Fortunately, the observation that it is extremely simple to distinguish noise-free phases in the EEG from artifact-related phases, merely by looking at the “EEG curve”, led to the idea of developing a device which makes use of the movement-related artifact bursts in the EEG signal.

The outcome of these considerations was a system which relies upon wilfully generated contractions of a single muscle of the system user. Consequently, it causes very little physical effort, while simultaneously being practicable and virtually *insensitive* to noise. The basic architecture of the system is illustrated in fig. 2. The three main building blocks of the system, dealing with the acquisition and the processing of the input signal, as well as the example target application (which in this case is a spelling device, but this is just one of numerous possibilities), are dealt with in the subsections below.

A. Data Acquisition

The EEG signal of the user of the system is recorded with the help of an EEG machine and fed into a computer (if necessary via an analog-to-digital converter), but instead of dealing with *genuine* brain-waves, the EEG signal is searched for (wilfully generated) muscle contraction artifacts. The reason for using *muscle contraction artifacts* as the information-carrying medium is that artifact detection can simply be done by inspecting the amplitude of the signal, since that is much lower in the artifact-free EEG.

By the way, the *EMG* signal of the subject can alternatively be used, since muscle contractions also result in larger EMG amplitudes (but the amplitude change in comparison to non-contraction phases is probably clearer in the EEG trace). Although muscle contractions are the main source of the EMG signal (and therefore are not regarded as *noise* or *artifacts* in this context), the resulting variations in the EMG amplitude are

²If the user wants to change the direction because he or she is heading for a staircase going down, it is critical for the system to service that request before the wheelchair reaches the first step...

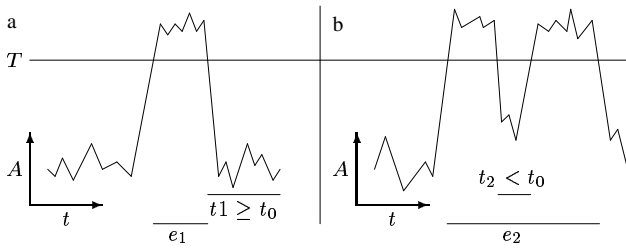


Fig. 3. Detectable Events

sufficiently characteristic. This means that everything stated in this paper about the detection of muscle contractions using an EEG machine equally applies to a *dedicated* EMG machine.

In order to further facilitate the detection of artifacts, the user chooses one particular muscle he or she can activate reliably and fast, and an electrode is placed directly above that muscle. Since the inspection of the amplitude refers to *one* time series only, this single electrode is already enough, if it is an *active* electrode (i.e. actively generating a voltage output). When using *passive* electrodes (which are much more common in conjunction with EEG/EMG analysis), two additional ones – e.g. respectively placed at each of the user’s ears – are needed: one reference and the ground electrode.

B. Signal Processing

The computer examines the amplitude A of the output of the primary acquisition device (i.e. an EMG or an EEG machine) and determines whether or not this amplitude exceeds a certain threshold T . This threshold, by the way, may vary from person to person, and it definitely depends on the kind of muscle that is used. Therefore, it is vital that the system can be individually adjusted at the beginning of every session. However, this “adjustment” can be done in about a minute, i.e. it is negligible.

If the amplitude of the signal exceeds the threshold once, as depicted in fig. 3a (which is similar to a single “mouse click”), the computer registers an e_1 event. In addition, if a second e_1 event is registered directly after the first (with a time interval of less than t_0 in between), those two events are regarded as a new one (which is similar to a “double click”). This new event – denoted e_2 – can be seen in fig. 3b. Actually, in order to make the system more intuitive for the user, an e_1 event is registered *at the onset* (the rising edge) of the event, and the rising edge shortly after an e_1 event is regarded as the onset of an e_2 event.

The detection of these two events – together with the corresponding reactions – is all that is needed to communicate with the computer. For example, it is possible to make the computer write characters, words and even entire letters³ if the input signal is processed in a clever way. This includes analyzing the input stream (so that only a sequence of e_1 and e_2 events – interrupted by non-event phases – remains) and deciding what to do if an event is generated by the user of the system.

C. Spelling Application

The working principle of the system is demonstrated with the help of a *spelling device* as an example target application. In

³In other words, a severely disabled person can write e-mails, i.e. have a “social life”, just by raising the eyebrow.

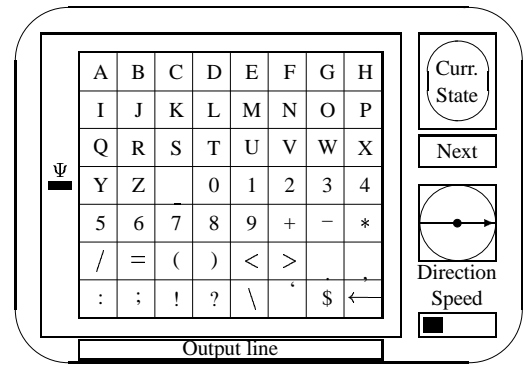


Fig. 4. “Spell-board” Screen

general, a spelling device involves the iterative selection of single characters, and the character selection algorithm used here is described in the following. The user of the system sees a “spell-board” containing

$$8 \times 7 = 56 \quad \text{characters} \quad C_1, C_2, \dots, C_{56}$$

arranged in a grid of 56 squares on a computer screen⁴ (fig. 4), where each square occupies 60×60 pixels on a 640×480 graph screen. The goal is to “move” an object Ψ on the screen, and to stop at the desired characters, one after the other. The last character (in the bottom right corner) serves as a correction symbol, so stopping at the corresponding square results in the deletion of the previously selected character (which is therefore considered to be falsely selected).

The object Ψ may be seen as a virtual vehicle that initially stands still. By generating an e_2 event, the vehicle starts moving straight – slow at first, but then speeding up until the next event is registered. If this is an e_1 event, the vehicle rotates to the left until the user generates another event. The time span between two events corresponds to a *state*, and the virtual movement can be described by the diagram of the simple *Finite State Machine* (FSM) in fig. 5.

By generating e_1 events, the user cycles through the STRAIGHT, LEFT, STRAIGHT', and RIGHT states, and he or she can stop in any state just by generating an e_2 event. When this happens, the computer determines the square corresponding to the current position of the vehicle on the screen and writes the associated character to the output line at the bottom of the “spell-board” screen.

On the right-hand side of the screen, the user is informed about the current and next states, the current direction and the current speed of the vehicle. This part of the screen is thus comparable to a dashboard in an automobile.

IV. IMPLEMENTATION

The system described above was realized using a PII/233 PC and an analog 8-channel EEG machine called “E8000” manufactured by *Schwarzer*. Actually, only one of those eight channels was used in the current implementation, since (as already

⁴Actually, this is the same computer that is responsible for the monitoring of the input signal, so the system does not require more than one computer.

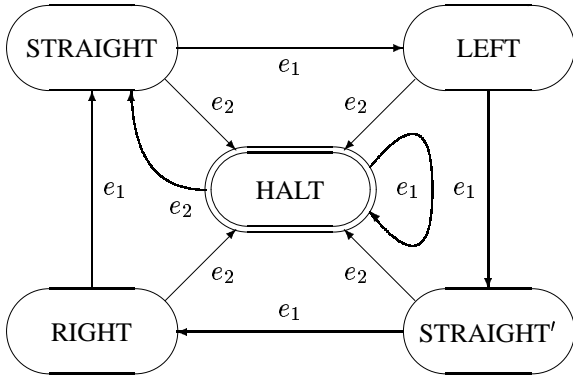


Fig. 5. Movement FSM

mentioned above) artifact-detection only requires a single signal.

The EEG output signal was digitized and sampled at 256 Hz and preprocessed by computing the mean amplitude of the data points belonging to the most recent 500 ms every 125 ms. In other words, the signal is converted into a one-dimensional time series $A(t)$, sampled at 8 Hz – this conversion process is illustrated in fig. 6. The time span from one inspection of $A(t)$ to the next is called a *step* (the reason for this being pretty obvious since A is actually a special kind of a *step function*).

The movement algorithm stores four properties characterizing the virtual object Ψ : its position $\vec{p} = (p_x, p_y)$, its current speed v (in *pixels per step* or *pps*), the direction of the last move (in the form of a unit vector $\vec{d}_0 = (d_{0,x}, d_{0,y})$ with $\sqrt{d_{0,x}^2 + d_{0,y}^2} = 1$) and the current state S of the movement FSM. These four properties are updated each time the value of $A(t)$ is inspected, which means that Ψ performs 8 movements per second.

The state S changes only after the registration of one of the events e_1 or e_2 in accordance with fig. 5. The instructions governing the updating of the other properties is dependent on the current value of S . The simplest case in this respect is of course the state $S = \text{HALT}$, where nothing is changed, i.e.

$$\begin{aligned} \vec{d}_0' &= \vec{d}_0 \\ \vec{p}' &= \vec{p} \\ v' &= v = 0. \end{aligned}$$

In the two STRAIGHT states, the vector determining the movement direction remains unchanged, and the speed – which is initialized with v_0 each time a state is entered – is constantly increased (by v_1), until a certain maximum v_{\max} (to be chosen empirically) is reached. The new position of the object Ψ results from the combination of the properties in the obvious way. This means,

$$\begin{aligned} \vec{d}_0' &= \vec{d}_0 \\ \vec{p}' &= \vec{p} + v \times \vec{d}_0 \end{aligned}$$

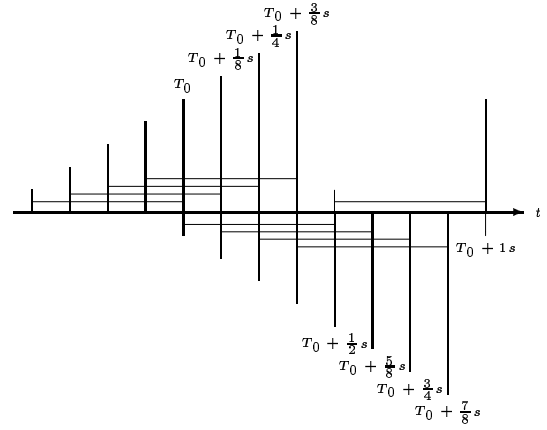


Fig. 6. 8 Steps/Moves per second

$$v' = \min(v + v_1, v_{\max}).$$

Finding an analytical formula representing a step in one of the turning states is somewhat more complicated. Let us suppose that, by definition, the speed remains constant – this makes the turning states somewhat *dual* to the straight states, since in the former ones, the speed is constant and the direction changes, whereas in the latter ones, the direction is constant while the speed increases:

$$v' = v = v_0.$$

In order to find the new direction that simulates a turn, one might think of an object Ψ' “being” at position \vec{p} moving forward a little (along the direction \vec{d}_0 currently stored) and then to the side (in orthogonal direction). The object Ψ' then arrives at a new position $\vec{p}_t = (p_{t,x}, p_{t,y})$. Normalization of the vector from \vec{p} to \vec{p}_t yields the new direction vector \vec{d}_0' .

It is extremely simple to find a vector that is orthogonal to a given one in two-dimensional vector space – it just amounts to exchanging the coordinates and inverting the sign of one of the coordinates. Consequently, $\vec{d}_{-1} = (-d_{0,y}, d_{0,x})$ and $\vec{d}_1 = (d_{0,y}, -d_{0,x})$ – which both are unit vectors orthogonal to \vec{d}_0 – correspond to left and right turns, respectively.

To determine \vec{p}_t , it therefore suffices to merely choose the magnitudes of the forward and sideward moves of Ψ' , i.e. the coefficients of \vec{d}_0 and \vec{d}_{-1} (for rotating to the left) or \vec{d}_0 and \vec{d}_1 (for rotating to the right).

The contribution of the sideward move should be proportional to the speed v_0 , and a value of $\zeta \times v_0$ with $\zeta = 0.05$ has empirically proven to be adequate. Similarly, a value of $\xi = 0.5pps$ was adequate for the amount of the forward contribution of Ψ' in the experiments reported below.

All these considerations result in the formulae

$$\begin{aligned} \vec{p}_t &= \vec{p} + \xi \times \vec{d}_0 + \zeta \times v_0 \times \vec{d}_{-1} \\ \vec{d}_0' &= \frac{\vec{p}_t - \vec{p}}{\sqrt{(p_{t,x} - p_x)^2 + (p_{t,y} - p_y)^2}} \end{aligned}$$

for the LEFT turn and

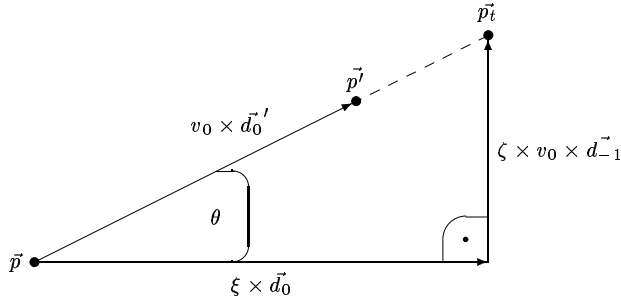


Fig. 7. LEFT turn

TABLE I
TYPICAL TURNING ANGLES

| v_0 | θ | t_c |
|--------|----------|--------|
| 0.5pps | 2.86° | 15.72s |
| 1.5pps | 8.53° | 5.28s |
| 2.5pps | 14.04° | 3.21s |

$$\begin{aligned} \vec{p}_t &= \vec{p} + \xi \times \vec{d}_0 + \zeta \times v_0 \times \vec{d}_1 \\ \vec{d}_0 &= \frac{\vec{p}_t - \vec{p}}{\sqrt{(p_{t,x} - p_x)^2 + (p_{t,y} - p_y)^2}} \end{aligned}$$

for the RIGHT turn.

It might be tempting to consider \vec{p}_t as the new position \vec{p}' , but the length of the vector $(\vec{p}_t - \vec{p})$ is not equal to $|v_0|$ (in general), and since we want the object Ψ to “move” with the “speed” v_0 , i.e. to “travel a distance” of $|v_0|$ pixels at each step, we arrive at the following formula:

$$\vec{p}' = \vec{p} + v_0 \times \vec{d}_0.$$

A graphical visualization of how a turning step is realized is presented for the example of the LEFT turn in fig. 7. The angle θ represents the amount the direction vector is turned (or rotated) at each step. It can be calculated using the arctan function:

$$\theta = \arctan\left(\frac{\zeta \times v_0}{\xi}\right).$$

Table 1 contains some typical values for θ (with $\zeta = 0.05$ and $\xi = 0.5pps$).

It is striking that θ seems to be almost proportional to v_0 . This oddity is due to the fact that the arctan function is almost linear for small arguments.

The turning step is iteratively executed as long as the state is not changed. In other words, if the system stays in the LEFT state for n seconds, the direction vector is turned to the left $8 \times n$ times by the angle θ (since there are eight steps per second), which corresponds to a total turning angle of

$$\Theta = 8 \times n \times \theta.$$

During the turning states, the virtual object Ψ describes a circle or, to be precise, a regular polygon⁵, where a full circle

⁵Actually, this is only correct if θ is a divisor of 360° – for general θ , the trajectory of Ψ is a sort of “thick polygon”

corresponds to a total turning angle of $\Theta = 360^\circ$. The time t_c the system has to remain in a turning state until Ψ has described a full circle depends (like θ) solely on the value of v_0 (for constant ζ and ξ). It is also listed in table 1.

It should be noted that all of the calculations mentioned above are carried out based on 64-bit floating point precision – merely for displaying, the position coordinates had to be converted to integer values.

V. EXPERIMENTAL RESULTS

To evaluate the system, two sets of experiments were carried out with a single subject: a 29-year-old male student, diagnosed with Friedreich’s Ataxia in 1986. The subject is obliged to use a wheelchair since 1988 and has only limited control over the upper extremities. Head and facial muscles are almost unaffected, though. Consequently, it was possible to choose the forehead as the site for placing the electrodes, and the subject was therefore able to operate the system simply by frowning.

A. The First Set of Experiments

The goal of the first set of experiments was to identify optimum (or at least *favorable*) values for v_0 and t_0 . The way to achieve this goal was twofold. First, a preselection process led to a few possible candidate values for the two constants. Second, the subject was asked to repeatedly “write” a fixed sentence with the system for each combination of the candidate values. The average time needed to “write” one character was taken as the performance measure.

In order to give the average values a certain validity, it was required to repeat the writing process for each combination sufficiently often. Therefore, it was decided to choose only three different values each for both constants, which limits the number of combinations to $3 \times 3 = 9$. The last section has shown that for $v_0 \in [0.5pps, 2.5pps]$, the “circle time” t_c ranges from 3s to 15s, and it was empirically determined that larger or smaller circle times would definitely *not* lead to optimum performance, so 0.5pps, 2.5pps and their average 1.5pps were chosen as candidate values for v_0 .

As for the “inter-click” time t_0 , the candidate values should be multiples of $\frac{1}{8}s$ since the amplitude curve is inspected eight times per second (see fig. 6). A value of $t_0 = 0.5s$ means that the subject needs to generate a second e_1 event within 500ms after the first (in order to make the virtual object start or stop), whereas $t_0 = 1.0s$ means that the subject has to wait an entire second after an e_1 event to change the moving state with another e_1 event (without the risk of unintendedly issuing an e_2 event). Empirical considerations showed that values outside the range $[0.5s, 1.0s]$ did not make too much sense, so 0.5s, 1.0s and their average 0.75s were chosen as candidate values for t_0 .

The subject’s job then was to repeatedly write the characters of the Fifth Commandment (fig. 8) as quickly as possible for each of the nine combinations. This was organized as follows. For each combination, two sessions with six trials each were performed by the subject. One trial comprised the selection of all of the 20 characters of fig. 8 in correct order. In addition, the subject was asked to remove any erroneous (“false”) characters

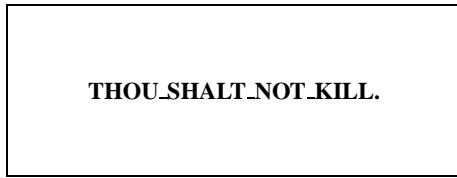


Fig. 8. The Fifth Commandment

by selecting the “←” symbol⁶. Therefore, the subject had to make a total of at least

$$9 \times 2 \times 6 \times 20 = 2160$$

correct selections. The 18 sessions were performed over the course of about four weeks, with no more than two sessions at the same day (this kept the strain on the subject’s forehead as low as possible). The time needed until the (initially empty) output line contained an exact copy of fig. 8 was recorded for each trial (with an inter-trial interval of at least one minute), and the results averaged over the twelve trials belonging to each one of the nine combinations are presented in table 2.

Before commenting further on these results, it seems appropriate to talk about the subject’s experience with “writing characters” with the system, in particular the adopted strategy when “moving” the virtual object from position \vec{x} to position \vec{y} . This is often similar to a particular pattern, which may be described as follows. After setting off at position \vec{x} in the STRAIGHT state with an e_2 event, the user changes to the LEFT state (by generating an e_1 event) and waits until the direction arrow on the right hand side of the screen points approximately in the direction of position \vec{y} . At that very moment, the user changes to state STRAIGHT’ to make the object stop turning and start moving straight. After that, when the object is near the goal, the user changes to the RIGHT state to make it stop moving straight. The user may then use the entire process over and over to “fine tune” the object’s position and eventually – when the object has reached position \vec{y} – the object may be halted by generating an e_2 event.

As can be seen in table 2, when operating the system with the combination $v_0 = 1.5pps$ and $t_0 = 0.75s$ – which was also the one the subject felt most comfortable with – the best average results were obtained. Moreover, all of the three combinations with $v_0 = 1.5pps$ performed better than the six other combinations. According to the subject’s impressions, this is due to two reasons.

First, with $v_0 = 0.5pps$, the user has to wait too long. If the user changes e.g. to the LEFT state, and the direction has turned enough (i.e. the turned direction vector has reached the angle corresponding to the direction the user wants the object Ψ to move), but t_0 has not elapsed yet (so another event would result in selecting a false character and not in a mere moving state change), the user has to wait the entire circle time (over 15s) – doing nothing – until the direction vector reaches the desired position again.

⁶It turned out that the number of errors in every trial was almost negligible, since the average number of errors was merely about two per trial, with many trials containing no errors at all.

TABLE II
RESULTS FOR THE FIRST SET OF EXPERIMENTS

| Param. v_0 | Param. t_0 | Time/trial | Time/character |
|--------------|--------------|------------|----------------|
| 0.5pps | 0.5s | 6.75 min. | 20.25s |
| | 0.75s | 6.70 min. | 20.10s |
| | 1.0s | 7.54 min. | 22.62s |
| 1.5pps | 0.5s | 5.94 min. | 17.82s |
| | 0.75s | 5.19 min. | 15.57s |
| | 1.0s | 6.69 min. | 20.07s |
| 2.5pps | 0.5s | 9.66 min. | 28.98s |
| | 0.75s | 7.05 min. | 21.15s |
| | 1.0s | 8.94 min. | 26.82s |

Second, for $v_0 = 2.5pps$ the turning angle θ is so large (over 14°) that the user often misses the right moment to make the object stop turning by generating an e_1 event, since the direction angle changes extremely fast and is therefore acceptable only during a very small time window.

Additionally, it can be seen that in every group of three combinations belonging to the same v_0 , the one with $t_0 = 0.75s$ performed better than the other two. The reasons for this may be deduced from the considerations that led to the decision of limiting t_0 to the interval $[0.5s, 1.0s]$. With $t_0 = 0.5s$, the subject needed two or three attempts to generate an e_2 event, since the time window of 500ms is too small. On the other hand, $t_0 = 1.0s$ led to a relatively large number of unintended e_2 events, since, in many cases, the desired direction angle – in one of the turning states – is reached *before* an entire second has elapsed, so the subject either waits until the object has reached that direction angle again (which means an additional t_c period) or (if impatient) produces an error.

B. The Second Set of Experiments

In an attempt to further reduce the average time per character, the movement FSM was slightly modified. Before, generating an e_1 event in the HALT state was simply discarded. After the modification, an e_1 event in state HALT resulted in a change of the orientation from FORWARD to BACKWARD and vice versa.

Formally, the modification results in a new state diagram (see fig. 9), where each state is doubled and – in particular – each of the four moving states is assigned an additional orientation attribute.

Since the virtual object Ψ does not really have a front or a back, the modification in fact simply amounts to rotating the direction angle by 180° :

$$\vec{d}'_0 = -\vec{d}_0.$$

The idea behind this is that the time needed to move Ψ from character C to the next character C' is very large if Ψ is “heading away” from C' (in the opposite direction) after the selection of the character C . The modification described above reduces this time, since it is now possible to choose between two (opposite) starting directions, so the user can make Ψ turn around more quickly (and doesn’t have to wait for $t_c/2$).

The usefulness of this idea was verified in a second set of experiments, where the subject was asked to repeat the two sessions with $v_0 = 1.5pps$ and $t_0 = 0.75s$, but this time the control

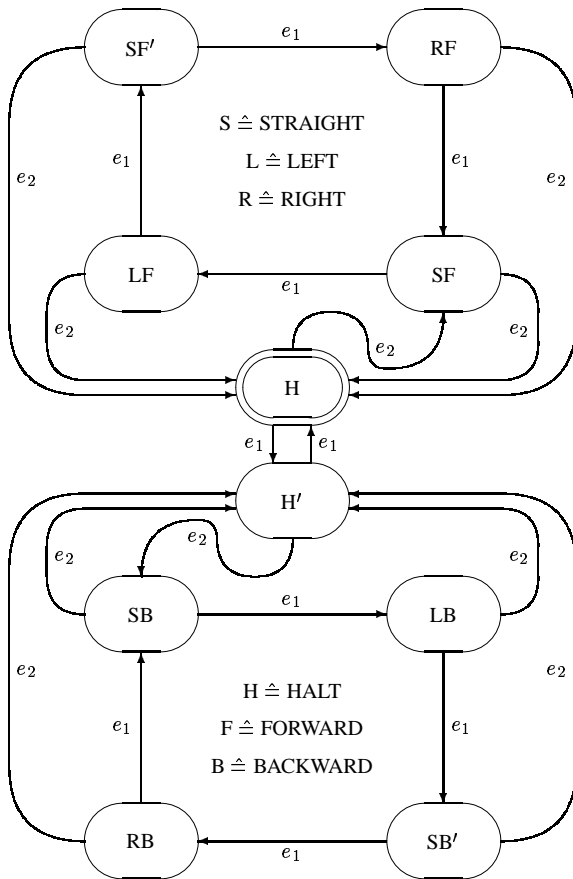


Fig. 9. Modified FSM

program was based on the modified FSM. The averages needed to write one character in each one of the twelve trials are presented in fig. 10. The overall average of $13.33s$ was indeed lower than the one with the original FSM.

Furthermore, as also shown in fig. 10, the maximum and minimum averages needed in those twelve trials were very close, which means that the result was extremely stable and reproducible.

It could be argued that the stability observed in the second set of experiments is the consequence of the training effect of the first, and that it depends on the particular task of copying the characters of the Fifth Commandment. While the training part of this argument may well be true – practicing surely helps to improve performance here – the rest is definitely not: following the second set of experiments, the subject was asked to “write” several *untrained* sentences (of similar length as the phrase in fig. 8, but containing different characters) with the same program parameters as in the second set of experiments – the results were analogous.

VI. CONCLUSIONS

A new kind of alternative input method has been introduced which allows interaction with a computer by contracting any muscle of one’s choice. Since the detection whether or not a muscle is contracted is almost unaffected by anything else but that muscle (e.g. any movement involving different muscles),

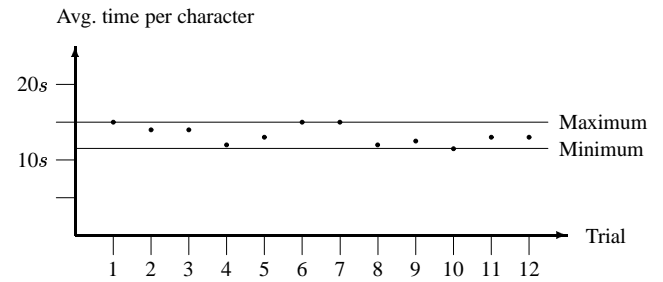


Fig. 10. The second set of experiments

the underlying system is very robust and practicable. The feasibility of the approach has been demonstrated with the help of two sets of experiments using a spelling device as an example target application. The device allows “typing” at a rate of at least 4–5 characters per minute, which makes it an effective alternative to the standard keyboard approach for disabled persons who cannot use their hands.

There is a wide range of possible applications of the system to be explored in the future. Some of these represent an aid assisting disabled people in using a computer, others constitute an alternative for everyone who wants to control something with the help of a computer.

An example for the first group of applications is the one detailed in this paper: a spelling device. It was demonstrated that it is possible to use the system as a substitution for the standard keyboard and mouse input devices. The results revealed that average time spans of $15s$ per character are not unusual, and it is conceivable to further facilitate and accelerate “writing” with a prolonged practice period and an appropriate word completion algorithm (so that it sometimes suffices to select only the first two or three characters of a word).

Combining the spelling device with a word processor offers the chance to compose e-mails, and the access of disabled individuals to the internet can additionally be enhanced by developing a mouse emulation device that uses the discrete state output of the system dealt with above to produce output commands for navigating in a web-browser.

Another assistive device is given by a control system for an electrical wheelchair. The application of the idea introduced here to a wheelchair control device has already been described elsewhere [31]. The prototype system detailed there gives wheelchair drivers a chance to steer the chair without the need to use the hands. In addition to the displacement of an actual wheelchair, it allows practicing without endangering the environment in a so-called *simulation mode* – which is actually almost identical to the application described in this paper involving a virtual object (as a substitute for the real wheelchair) that can be moved around on a computer screen. As one might expect, the prototypical implementation is easy to use, effective, robust, and secure – e.g. it even allows talking (while driving the wheelchair). In addition, the control software turns out to be very small and the hardware to be extremely inexpensive, so the system represents a practicable alternative compared to the traditional joystick steering method.

Furthermore, since any device based on the described input method can be operated in an easy and effortless way, is almost independent of any undesired interference, and requires

between one and three electrodes only, it is imaginable that even able-bodied subjects use the method in a number of applications. Examples include a phone headset allowing to dial by “frowning” or an “eyebrow-operated” infrared remote control for the home entertainment center.

It can be concluded that there seems to be a huge number of possible uses of the method presented here. Future tasks refer to the identification of those possibilities, the realization of promising applications, and the testing of their usefulness.

ACKNOWLEDGMENTS

The authors wish to thank Mr. Kandler from *NeuroKard* for supplying the EEG device needed (at no charge!) and for giving useful hints on its operation.

REFERENCES

- [1] H. Callender, W. Pruehsner, and J. D. Enderle, “LED directed, headswitch controlled, motorized wheelchair,” in *Proceedings of the IEEE 27th Annual Northeast Bioengineering Conference*, 2001, pp. 85–86.
- [2] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kübler, J. Perelmouter, E. Taub, and H. Flor, “A spelling device for the paralyzed,” *Nature*, vol. 398, pp. 297–298, 1999.
- [3] N. Birbaumer, “Slow cortical potentials: Plasticity, operant control and behavioral effects,” *The Neuroscientist*, vol. 5, pp. 74–78, 1999.
- [4] J. Perelmouter, B. Kotchoubey, A. Kübler, E. Taub, and N. Birbaumer, “Language support program for thought-translation-devices,” *Automedica*, vol. 18, pp. 67–84, 1999.
- [5] A. Kübler, B. Kotchoubey, T. Hinterberger, N. Ghanayim, J. Perelmouter, M. Schauer, C. Fritsch, E. Taub, and N. Birbaumer, “The thought translation device: a neurophysiological approach to communication in total motor paralysis,” *Exp. Brain Res.*, vol. 124, pp. 223–232, 1999.
- [6] J. Vidal, “Toward direct brain-computer communication,” *Annu. Rev. Biophys. Bioeng.*, pp. 157–180, 1973.
- [7] O. Fukuda, T. Tsuji, and M. Kaneko, “Pattern classification of EEG signals using a log-linearized gaussian mixture neural network,” in *Proceedings of the IEEE International Conference on Neural Networks*, 1995, vol. 5, pp. 2479–2484.
- [8] D. Steuer, B. Schack, G. Grieszbach, and W. Krause, “Classification of human cognitive processes by the use of an improved neural backpropagation network,” in *Proceedings of the IEEE International Conference on Neural Networks*, 1995, vol. 5, pp. 2495–2500.
- [9] J. R. LaCourse and E. Wilson, “Brainiac: A brain computer link,” in *Proceedings of the Instrumentation and Measurement Technology Conference of the IEEE*, 1995, pp. 587–592.
- [10] D. Lisogurski and G. E. Birch, “Identification of finger flexions from continuous EEG as a brain computer interface,” in *Proceedings of the 20th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 1998, vol. 20, pp. 2004–2007.
- [11] M. Polak and A. Kostov, “Training setup for control of neural prosthesis using brain-computer interface,” in *Proceedings of The First Joint BMES/EMBS Conference: Serving Humanity, Advancing Technology*, 1999, p. 446.
- [12] E. Donchin, K. M. Spencer, and R. Wijesinghe, “The mental prosthesis: Assessing the speed of a P300-based brain-computer interface,” *IEEE Transactions On Rehabilitation Engineering*, vol. 8, no. 2, pp. 174–179, 2000.
- [13] J. R. Wolpaw, N. Birbaumer, W. J. Heetderks, D. J. McFarland, P. H. Peckham, G. Schalk, E. Donchin, L. A. Quatrano, C. J. Robinson, and T. M. Vaughan, “Brain-computer interface technology: A review of the first international meeting,” *IEEE Transactions On Rehabilitation Engineering*, vol. 8, no. 2, pp. 164–173, 2000.
- [14] T. Felzer and B. Freisleben. “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*, Las Vegas, Nevada, USA, 2002, vol. 2, pp. 329–335, CSREA Press.
- [15] J. R. Wolpaw, D. J. McFarland, G. W. Neat, and C. A. Forneris, “An EEG-based brain-computer interface for cursor control,” *Electroencephalography and clinical Neurophysiology*, vol. 78, pp. 252–259, 1991.
- [16] J. R. Wolpaw and D. J. McFarland, “Multichannel EEG-based brain-computer communication,” *Electroencephalography and clinical Neurophysiology*, vol. 90, pp. 444–449, 1994.
- [17] J. Kalcher, D. Flotzinger, S. Göllly, C. Neuper, and G. Pfurtscheller, “Graz brain-computer interface (BCI ii),” in *ICCHP '94 – Computers for Handicapped Persons*, W. L. Zagler, G. Busby, and R. R. Wagner, Eds., 1994, pp. 170–176, Springer-Verlag.
- [18] B. O. Peters, G. Pfurtscheller, and H. Flyvbjerg, “Mining multi-channel EEG for its information content: An ANN-based method for a brain-computer interface,” *Neural Networks*, vol. 11, pp. 1429–1433, 1998.
- [19] C. W. Anderson, S. V. Devulapalli, and E. A. Stolz, “Determining mental state from EEG signals using neural networks,” *Scientific Programming – Special Issue on Applications Analysis*, vol. 4, no. 3, pp. 171–183, 1995.
- [20] C. W. Anderson and Z. Sijerčić, “Classification of EEG signals from four subjects during five mental tasks,” in *Solving Engineering Problems with Neural Networks: Proceedings of the Conference on Engineering Applications (EANN'96)*, A. B. Bulsari, S. Kallio, and D. Tsaptsinos, Eds., Turku, Finland, 1996, pp. 407–414, Systems Engineering Association.
- [21] E.-A. Degermann, R. Dahlberg, D. Wallen, E. Björklund, and D. Lundman, “Ergonomic and technical evaluation of an eye-controlled computer with ‘eyegaze’,” *Work*, vol. 5, pp. 213–221, 1995.
- [22] J. J. Tecce, J. Gips, C. P. Olivieri, L. J. Pok, and M. R. Consiglio, “Eye movement control of computer functions,” *International Journal of Psychophysiology*, vol. 29, pp. 319–325, 1998.
- [23] K. S. Park and K. T. Lee, “Eye-controlled human/computer interface using the line-of-sight and the intentional blink,” *Computers & Industrial Engineering*, vol. 30, no. 3, pp. 463–473, 1996.
- [24] O. A. Alsayegh, “EMG-based human machine interface system,” in *Proceedings of the 2000 IEEE International Conference on Multimedia and Expo (ICME 2000)*, vol. 2, pp. 925–928.
- [25] H.-J. Park, S.-H. Kwon, H.-C. Kim, and K.-S. Park, “Adaptive EMG-driven communication for the disability,” in *Proceedings of the First Joint BMES/EMBS Conference: Serving Humanity, Advancing Technology*, 1999, vol. 1, p. 656.
- [26] Y.-H. Tarn, G.-C. Chang, J.-S. Lai, and T.-S. Kuo, “Design of the human/computer interface for human with disability using myoelectric signal controlled,” in *Proceedings of the 19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 1997, vol. 5, pp. 1909–1910.
- [27] A. M. Junker and C. R. Berg, “The Cyberlink interface: Hands free brain-body actuated control for augmentation and enhancement of human computer interaction,” in *Proceedings of the 2000 Technology and Persons with Disabilities Conference of the Center on Disabilities of the California State University at Northridge*.
- [28] A. M. Junker, J. R. Wegner, and T. Sudkamp, “Cyberlink brainfingers for people with no means of access, NIH study results,” in *Proceedings of the 2002 Technology and Persons with Disabilities Conference of the Center on Disabilities of the California State University at Northridge*.
- [29] W. T. Nelson, L. J. Hettinger, J. A. Cunningham, M. M. Roe, M. W. Haas, and L. B. Dennis, “Navigating through virtual flight environments using brain-body-actuated control,” in *Proceedings of the 1997 IEEE Annual International Virtual Reality Symposium*, 1997, pp. 30–37.
- [30] C. K. Ho and M. Sasaki, “Brain-wave bio potentials based mobile robot control: Wavelet-neural network pattern recognition approach,” in *Proceedings of the 2001 IEEE International Conference on Systems, Man, and Cybernetics*, 2001, vol. 1, pp. 322–328.
- [31] T. Felzer and B. Freisleben. “HaWCoS: The “hands-free” wheelchair control system,” in *ASSETS 2002 – Proceedings of the Fifth International ACM SIGCAPH Conference on Assistive Technologies*, Edinburgh, Scotland, 2002, pp. 127–134, ACM Press.