ORIGINAL RESEARCH

An EMG keyboard for forearm amputees

Wenwei Yu, Ryu Kato, Fukuda Fabio, Hiroshi Yokoi, Yukinori Kakazu

Complex Engineering Department, Hokkaido University, Sapporo, Japan

Abstract: A high-efficiency, easy-to-use input device is not only important for data entry but also for human–computer interaction. To date, there has been little research on input devices with many degrees of freedom (DOF) that can be used by the handicapped. This paper presents the development of an electromyography (EMG)-based input device for forearm amputees. To overcome the difficulties in analysing EMG and realising high DOF from biosignals, the following were integrated: (1) an online learning method to cope with nonlinearity and the individual difference of EMG signals; (2) a smoothing algorithm to deal with noisy recognition results and transition states; and (3) a modified Huffman coding algorithm to generate the optimal code, taking expected error and input efficiency into consideration. Experiments showed the validity of the system and the possibility for development of a quiet, free-posture (no postural restriction) input device with many DOF for users, including forearm amputees.

Keywords: input devices, EMG analysis, Huffman coding, online learning

Introduction

A high-efficiency, easy-to-use input device is not only important for data entry but also for human–computer interface (HCI). This interaction has been attracting much attention recently with the increased use of human-oriented mechatronic devices.

Input can be considered as a process of putting information into machines by making selections from a set of provided choices. Using this approach, input devices can generally be classified into 4 types based on the selection method and choice presentation. These types are described in Table 1.

When using a normal keyboard (an example of the 'physically extended' category), a person selects the correct character for pressing by moving on a set of spatially, physically extended key switches. Mouse devices belong to another spatially extended type, 'cursor select', but the icons to be chosen are usually extended virtually on a graphics layout. Input devices of most brain–computer interfaces (BCI) belong to this type (Birbaum et al 1999). The spatial moves for these two types need some kind of feedback to guarantee the reaching of the correct position to finish selection. Cursor select types need more feedback information, since moving the cursor by a roller to the target generally needs more than one action. Moreover, cursor moving relies on the state of the device, eg roller and mat conditions. Conversely, 'temporally extended' types, eg Morse code telegraphs, need a sequence of actions to input a character; however, since no spatial feedback is needed, despite the few degrees of freedom (DOF) choice, it can realise high-speed input. With practice, even with Morse code, one can reach an average of 30 wpm (words per minute), which is comparable to that of usual keyboards' 40–50 wpm.

Several companies have released effective voice input software packages; however, speech recognition-based data entry is still developing. A speech recognition system is large compared with other data-entry methods, as there needs to be a phoneme database, a word database and a sentence database prepared and installed. In addition, recognition accuracy is still not satisfactorily high. Moreover, a quiet and exclusive environment is usually the precondition of using such a data-entry method.

Table 1 Comparison between different types of input devices

Input device type	*Selection method*	*Choice DOF*	*Choice presentation*	*Choice type*
Physically extended eg keyboard	Move + key pressing	101	Spatially, physically extended	Character
Cursor select eg mouse	Move + click	X, Y, click (3)	Spatially extended in screen	Depends on setting
Coding eg Morse code	Key pressing	Click or not (1)	Temporally extended, encoding	Character
Speech recognition	Voice	Number of phoneme	Signal processing	Phoneme

Correspondence: Wenwei Yu, Complex Engineering Department, Hokkaido University, North 13, West 9, Sapporo 060-8628, Japan; tel +81 11 706 6446; fax +81 11 708 5188; email yu@complex.eng.hokudai.ac.jp

The basic evaluation criteria of an input device includes input DOF, input speed and ergonomic factors such as posture, effects on surroundings and the space occupied. Regarding ergonomic factors, many alternative keyboards have been recently developed, improving ease of typing, posture during typing, or space saving (Lueder and Grant 1997). Senseboard® (http://www.senseboard.com/index.php) realises a space-free input by capturing the motion of hands and fingers. Virtual Keyboard (VKB) (http://www.vkb.co.il/) uses an optical image projection technology combined with a detection module to achieve a small size and low power consumption input method.

However, there is little research on input devices with large input DOF that can be used by the handicapped. Barreto (1999, 2000) gave a practical electromyography (EMG) interface for the handicapped, but no consideration was shown for how to construct high DOF input devices. The Click-N-Type virtual keyboard (http://www.lakefolks.org/cnt/) was motivated by the needs of quadriplegic users, and was developed for people with limited muscle control. It also belongs to the cursor select type.

This research attempted to develop an EMG-based input device with large DOF for forearm amputees. EMG signals were detected from the upper forearms of amputees where, for those who had had prompt treatment and rehabilitation, muscle function remains after amputation. For forearm amputees, this device will not only help them restore part of their working ability, improving their quality of life, but will also provide exercise for their remaining muscles. Moreover, technologies of the input device can also be applied to non-amputees to improve some of the ergonomic factors of input.

As physical key touch or press is impractical for amputees, and EMG signals detected from a few channels cannot supply a sufficient action pattern repertoire, the strategy needed to: (1) take as many stable elementary action patterns as possible from a few channels of EMG signals so that the temporal extension could be reduced to a minimum; (2) treat the action patterns recognised from EMG signals as basic digits, eg '0' and '1' of the binary system; and (3) encode the elementary action patterns to express characters, considering the possible recognition errors and input efficiency. Therefore, the input device attempted was the 'coding' type (see Table 1). In such a system, users need to memorise a codebook developed for them. Comparatively high speed, quiet input with large DOF in natural postures are features of the system.

EMG signals have been used in keyboard research (Smith and Cronin 1992; Martin et al 1996), although mainly as a rough, relative measurement of force exertion and fatigue.

To use EMG signals to analyse forearm motions, the following difficulties in EMG processing should be considered (Yu et al 2002):

- It is difficult to specify a certain action intention from the superposition of multiple potentials, since the electric potentials of activated muscles are affected by various nonlinear elements such as fat and tissue.
- The interface using EMG should be individual-adaptive, since motion patterns of human beings are subject to a wide range of individual variations.
- EMG analysis should be able to trace the alternation, since subjects' characteristics vary through time due to environmental influence, muscle fatigue and physical states of subjects, especially in the initial stage of learning.

To overcome these problems, we integrated two approaches into our system: a machine-learning approach and an online approach. Since it is difficult to write general recognition rules for different users, an artificial neural network (ANN) model was used to learn and adapt to the individual characteristics. The online process was employed to receive teaching signals from operators and form an internal evaluation, which accelerates the learning and enables the controller to trace the non-stationary factor. Previous research shows that it is possible to distinguish 6–10 forearm action patterns for prosthetic hand control from two to three channels of EMG within 10–15 min (Nishikawa et al 2002).

One objective of this research was to apply the online method to EMG analysis for realising EMG input with large DOF. However, one feature of input action gives the particular requirement that is different with EMG analysis for prosthetic hand control. That is, in the case of prosthetic hand control, without considering delay, it is the signals in the steady stage of the muscle potential that should be used. Namely, each intended action could be taken as one stationary state. A short sequence of transitional state outputs or noisy outputs will not show too much bad influence on the prosthetic hand control, since they will be overwritten by prevailing right action sequences. Conversely, due to the continuous nature of input action, transitional states and unsteady states are unavoidable and will cause erroneous characters to be recognised. Therefore, mechanisms that can deal with the noisy and transition states should be

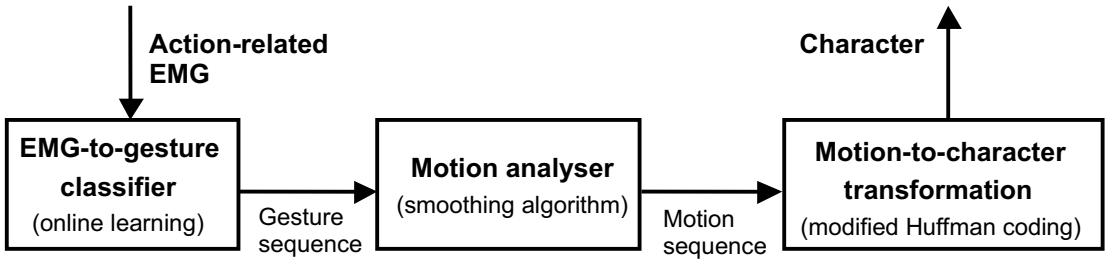

Figure 1 System overview.

incorporated. Moreover, the efficiency of input requires fewer expected actions for text input, with lower overall expected error rate.

The basic assumption of this study is that instead of treating forearm motions as completely continuous ones, each motion can be taken as a static gesture state enfolded by several transitional states. The underlying fact supporting this assumption is that smooth prosthetic hand control can be realised by static gesture recognition from forearm EMG signals, ie the static period prevails over an overall motion process.

According to the above analysis, the following 3 methods were integrated into the EMG keyboard system:

1. the backpropagation-based online learning method to cope with nonlinearity and individual difference;
2. a smoothing algorithm to deal with the noisy recognition results and transition states;
3. a modified Huffman coding algorithm to generate the optimal code considering expected error and input efficiency.

The details of the EMG analysis system, the smoothing algorithm, as well as the modified Huffman coding algorithm will be described in the following section.

Method

The overall system was constructed by connecting the 3 methods mentioned above (see Figure 1). The first is an EMG-to-gesture classifier, which receives raw EMG as input, and outputs sequences of gesture labels attached by transitional and noisy states. In this paper, the term 'gesture' denotes the part of a forearm action that can be taken as static, while the term 'motion' denotes the whole process of a continuous forearm action. The online learning method was employed to realise the classifier. The second part is a motion analyser, which analyses gesture sequences – raw recognition results – processing single spot noise and transitional states to output motion label sequences. The third is motion-to-character transformation, realised by a modified Huffman coding algorithm.

Online learning method

The online learning method was proposed for analysing the physiological signals conforming to an individual's characteristics. Figure 2 shows the composition of the EMG analysis system, which is composed of three main parts. The first is a feature-extracting part, which extracts necessary and sufficient information from source signals. The second is a data management part, which generates and maintains (does data management for) a training set for the adaptive recognition part from the teaching signals sent by operators. The third is a recognition part that determines operations according to current feature vectors and is implemented by an ANN in this edition. These three parts (ie feature extracting, teaching, learning and recognition)

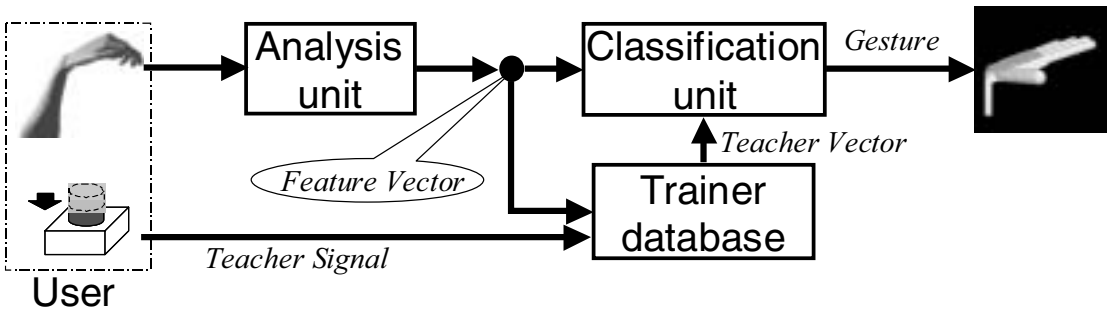

Figure 2 Composition of an online EMG-to-gesture classifier.

work simultaneously, ie if users feel that the system is not working so well, they can give a teach signal to cause the partial reconstruction of the recognition part; meanwhile, the system should keep recognising and learning based on the current function (Nishikawa 2002).

Feature vector

A fast Fourier transformation (FFT)-based analysis was employed, and the spectrum generated was further smoothed. The transformation can be briefly expressed by the following equation:

$$G\left(\frac{n}{N'\tau}\right)=\sum_{k=0}^{N'-1}\left\{\tau\cdot f(k\tau)e^{-i2\pi kn/N'}\right\} \quad (1)$$

where N' is the frame size for Fourier transformation, and τ is the time constant. After the transformation, the frequency curves are further smoothed:

$$G'\left(\frac{n}{N'\tau}\right)=\frac{1}{2w+1}\sum_{k=n-w}^{n+w}G\left(\frac{k}{N'\tau}\right) \quad (2)$$

where w denotes a range for smoothing. Features were then evenly selected from whole components, which would be $G'(200)$, $G'(300)$ and $G'(400)$ for each channel.

Backpropagation neural network

An ANN was used for the recognition part. In this edition, a three-layer backpropagation neural network (BPNN) was used for adaptation (Aihara and Douya 1993). The output of each neuron was calculated by the following equation:

$$x_i^m=f(u_i^m)=f\left\{\sum_{j=1}^{n_{m-1}}w_{ij}^m x_j^{m-1}\right\} \quad (3)$$

$$(m=2,3;\ i=1,\ldots,n_m)$$

where n_m is the number of mth layer's unit, u_i^m the input, x_i^m the output of ith unit of the mth layer, and w_{ij}^m is the weight between the (m–1)th and mth layers. Each unit of the network used the sigmoid activation $f(x)=1/(1+e^{-\alpha x+\beta})$, where α and β are constants that determine the transition of the neural unit. In this experiment, $\alpha=1$ and $\beta=1$. The purpose of the learning was to minimise the energy function E, defined by $E=\frac{1}{2}\sum_{i=1}^{n_3}(x_i^3-d_i)^2$, where d_i is the target value for each training pair i. The update of weights was based on the backpropagation (Aihara et al 1993), which can be expressed as:

$$\delta w_{ij}^m=-\mu\frac{\partial E}{\partial w_{ij}^m}=-\mu\frac{\partial E}{\partial x_i^m}\frac{\partial x_i^m}{\partial u_i^m}\frac{\partial u_i^m}{\partial w_{ij}^m}=\mu f'\left(u_i^m\right)x_j^{m-1} \quad (4)$$

$$(m=2,3)$$

Note that f' denotes the differential calculus of function f, and μ is a parameter denoting the learning speed.

Smoothing

The transitional states enfolding one certain static state are somewhat regular; therefore, if they were made use of, they may contribute to the more accurate gesture or motion recognition. However, this needs systematic study to first find the underlying regularity. In this research, the emphasis was on the static gesture part, simply removing the transitional states and noisy recognition results to extract motion information from raw recognition results.

Three heuristic rules were employed to cope with the transitional states and noisy recognition results (Figure 3). These 3 rules are executed serially; ie the 3 rules scan a label sequence 3 times.

Let '1' be a stop bit, let '2–7' be gesture labels generated by the EMG-to-gesture classifier, and let '*' be a mask denoting a temporary symbol. The objective of the first rule is to remove noisy recognition results. Since most physiological signal sources can be described as stochastic processes interwoven with each other, mis-recognitions are almost unavoidable. The first scan replaces those single spots with a temporary '*' mask. The scan by rule 2 is to keep motion continuity. If two continuous sequences with the same label are separated by a '*' mask, the single '*' is replaced with a label the same as its neighbour. Or, if the number of '*' is larger than 1, then each '*' is replaced by a stop bit '1'. The last scan by rule 3, a rule contributing to motion stability, is used to extract motion labels from a sequence of gesture labels. The sequence of identical labels with a length longer than threshold *Th* is extracted as a motion of the same label. All the others are treated as stop

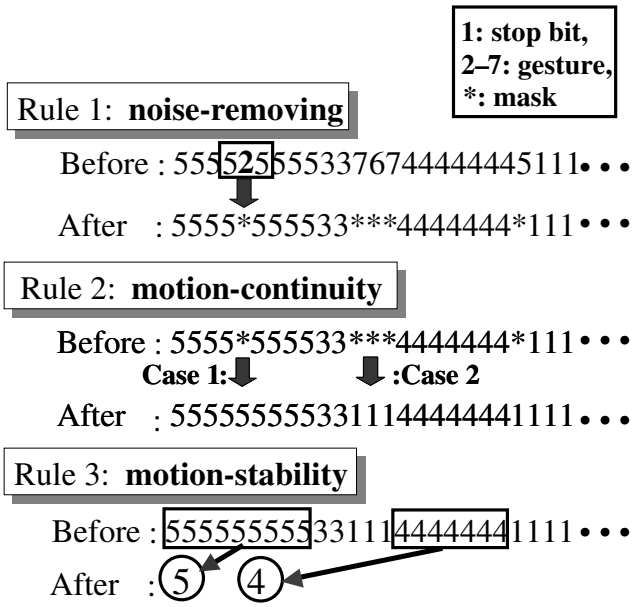

Figure 3 Three heuristic smoothing rules.

bits. *Th* decides the trade-off between promptitude and accuracy of input. In the experiment described, *Th* was set to 5.

A modified Huffman coding algorithm

Huffman coding is a technique that attempts to reduce the average code length used to represent the symbols of an alphabet. To reduce the average code length, symbols are allowed to be of varying lengths, and the shortest codes are assigned to the most frequently used symbols.

This meets the requirement of the EMG keyboard, which is to input the required content using as few forearm motions as possible. The difference is that, in addition to the requirement of fewer motions, lower overall expected error rate is an important condition. The rudimentary action patterns are not completely guaranteed due to the stochastic nature of signal sources and difficulties in signal processing. Therefore, a code with fewer expected motions for a character set and lower expected error rate was needed. See Algorithm 1 for the modified Huffman coding algorithm.

After a Huffman code tree is generated, the code for each symbol may be obtained by tracing a path to the symbol from the root of the tree. In the case of a binary tree, a '0' is assigned for a branch in one direction, and a '1' is assigned for a branch in the other direction. For example, a symbol that is reached by branching right twice then left once may be represented by the pattern '001'.

Step 0: Sort M according to R, to a buffer B_r

Step 1: Sort K according to P, to a buffer B_p

Step 2: Select n_m keys with lower p

Step 3: Integrate the keys to a new node g

Set $g > n_k$, $p_g = \Sigma_j^{nm} p_j$

Insert node g into B_p

Assign m of B_r orderly to the keys

Step 4: If $num(B_p) > n_m$, goto Step 2

Where:

character set:	$K\ (k_1, k_2, \ldots, k_{nk})$	
probability of K:	$P\ (p_1, p_2, \ldots, p_{np})$	$n_p = n_k$, $\Sigma_i p_i = 1$
motion set:	$M\ (m_1, m_2, \ldots, m_{nm})$	
correct rate of M:	$R\ (r_1, r_2, \ldots, r_{nr})$	$n_m = n_r$

Algorithm 1 The modified Huffman coding algorithm. In steps 0 and 1, the character and motion sets are sorted according to the probability of K and recognition correct rate of motion M, respectively, to code the keys with higher appearance probability, achieving more reliable motions and shorter code length.

Experimental setting

The experiment setting is illustrated in Figure 4. Two EMG electrodes (Model DE-02.3H, Delsys) were placed on the ulna and radius side within a belt zone 15–20 cm above the wrist joints, respectively. The amplifier was handmade. The A/D card used was CONTEC AD12-8 (PM) (input range –10 V to +10 V, resolution 12 bit). The sampling rate used for each channel was 1600 Hz.

A laptop computer (Panasonic CF-A2, 600 MHz) was used to realise the 3 processing components. The teaching signals were given through the keyboard of the laptop. Operators can see both the EMG recognition results and CG (computer graphics) hand motion displayed on the laptop monitor. The subject gestures are illustrated by icons of CG hand used in the experiment in Figure 5.

The parameters for feature extraction were decided by trial-and-error: $N' = 256$, $\tau = 0.000625$, and smoothing parameter $w = 16$.

Ordinarily, the extracted feature vectors were used as the inputs of the recognition part to decide operation commands. However, when a teaching signal was received, the feature vector at that moment would also be added to the training set, with a teaching label given by the operator.

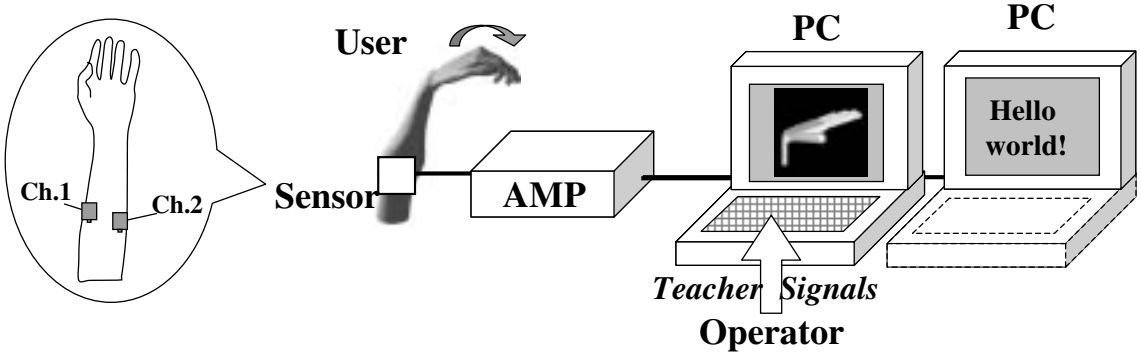

Figure 4 Experiment setting. The subject gestures are illustrated in Figure 5 by icons of CG (computer graphics) hand used in the experiment.

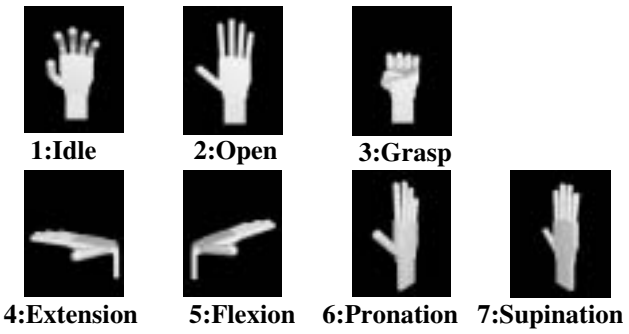

Figure 5 Subject gestures.

Each training set contains a 22-dimensional feature vector (11 for each channel) and a teaching label pair given by the operator. The BPNN used to realise recognition consists of 3 layers with an input layer neuron number of 22, a hidden layer neuron number of 22, and an output layer neuron number of 7 (corresponding to the 7 gestures shown in Figure 5).

Results and discussion

In this section, the aforementioned 3 processes used to construct an EMG keyboard are at first separately investigated and discussed, and, finally, the overall performance of the whole system is tested by a text-input experiment.

Online learning for single gestures

Since the detecting positions change every time the sensors are mounted and, also, individual characteristics change with time, the feature vectors of both channels are inconstant. However, if the recognition rate is sufficiently high, there should be differences in feature space between all the motion patterns. To give a visual illustration of how the gestures were differentiated, averaged feature vectors of all 6 gestures in one trial are shown in Figure 6. As shown in the figure, the features of two channels take effect alternately, so at least from the averaged feature vectors, these 6 gestures are distinguishable. Kato et al (2002) investigated the differentiability of physiological data, but it is beyond the main topic of this paper.

The effectiveness of learning was investigated using accuracy tests. In an accuracy test, users were required to control the CG hand to a gesture instructed on the monitor for a period of 15 s. The results were then recorded and used to calculate an accuracy rate.

Figure 7 shows the recognition results for 10-trial online learning. The accuracy rates of 6 gestures were averaged and are denoted by bars in the figure with their deviations. Although the standard deviation fluctuated with the trials, the overall performance kept around 90%. According to our experience on myo-controlled prosthetic hand control, this average accuracy rate is sufficient for static control.

Smoothing raw recognition results to motion patterns

Due to the existence of noisy states and transitional states, the high accuracy rates of the static gesture recognition do not mean a high success rate of text input. This can be made clear by a continuous action experiment.

In the continuous action experiment, each gesture was paired with another gesture, with 1 idle action as the stop bit between them. For example, '213' means open (2) and grasp (3), with a middle idle. 6×6 pairs of gestures were tested. Each pair was repeated 10 times. By using the same classifier obtained in the first stage, accuracy tests were carried out again.

Figure 8 shows one of the executions of a gesture pair '514'. Thin lines denote raw recognition results, and thick lines denote smoothed results. Single-spot noisy outputs,

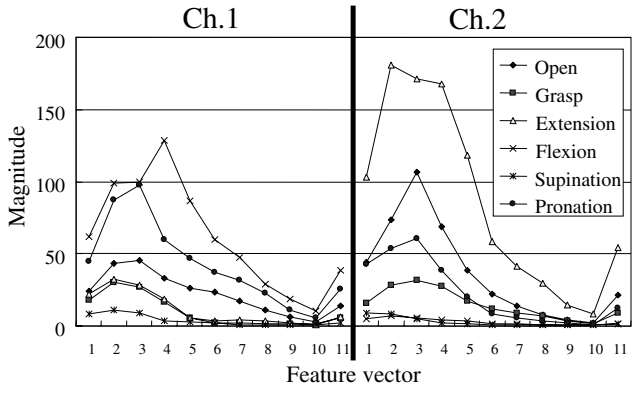

Figure 6 Feature vectors of two channels of EMG.

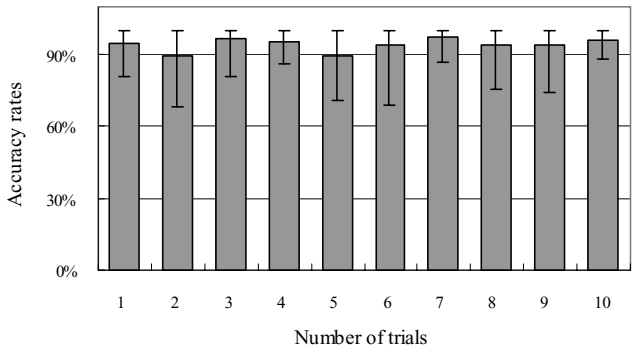

Figure 7 Recognition results for 10 trials. Each bar represents the average value over 7 gestures recognised and their deviations.

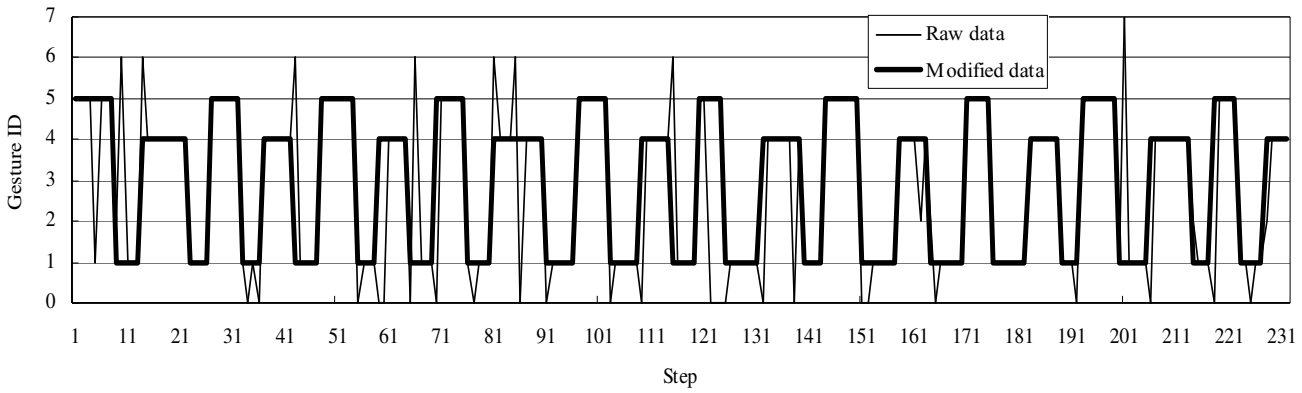

Figure 8 Raw recognition results and smoothing (for 514).

the transitional states, are expressed by the inconsistencies of the up or down edges of the rectangular wave. By knowing the gestures being executed, the raw results can be smoothed and modified into a reasonable motion sequence. However, in the case of real text input, the gesture being executed would be unknown, so that erroneous recognition existed even after the smoothing, which needs to be considered in the key code generation stage. By smoothing without knowing the gestures being executed, the average (averaged over $6 \times 6 \times 10$ executions) accuracy rates (see Figure 9) show that the accuracy rates for each gesture drop by about 10%.

Alphabetic keystroke occurrence frequency and Huffman coding

Keystroke occurrence frequencies differ from one occupation category or research field to another. Although there are many keystroke statistics in certain specialist fields, such as in Wall Street-related journals and some scientific journals, it is necessary to investigate the characteristics of input of each individual, including the edit keys, such as the backspace and arrow keys. Fortunately, it is not too difficult to do this. As an example, we investigated the keystroke occurrences by recording all the key code values while a 33-year-old subject input a piece of text, which is the Preface (p xi–xviii) of the book *Bioinformatics – The Machine Learning Approach* edited by Pierre Baldi and Soren Brunak. The text includes 35 paragraphs, 3120 words and 19 035 characters (22 235 characters if SPACE included). The occurrence frequency can be calculated using this data (see Figure 10). Details can be found in Appendix 1.

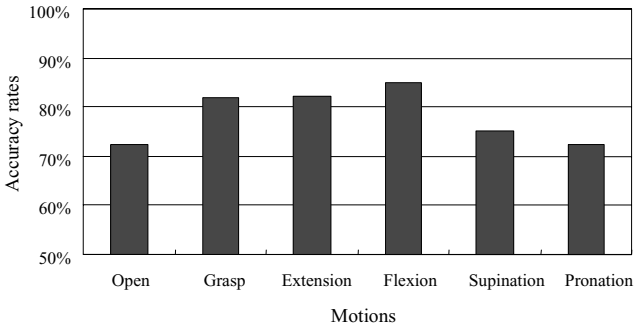

Figure 9 Accuracy rate for continuous motion recognition.

Given the motion accuracy rates and the occurrence frequencies, a Huffman code can be generated according to the algorithm shown in Algorithm 1. Motion codes of some high-frequency characters are shown in Figure 10. Details of the codebook can be found in Appendix 2.

Table 2 gives a comparison between the modified Huffman coding and 3 other fixed-length codes, in terms of motion number needed and expected average accuracy, for the example text in Appendix 1. Regarding the 3 fixed-length codes, description of 81 symbols needs 3 digits using 5 or 6 motion patterns, and 4 digits using 3 or 4 motion patterns. Their codes are generated by assigning the motion patterns with higher correct rate to smaller numerals. For example, in the case of 5 motions 3 digits, flexion(5), with a highest correct rate of 0.85, will be assigned to '0', and extension(4), with a second high correct rate of 0.83, will be assigned to '1'. The item 'hit count' expresses the number of the figure, ie the motion used in order to input the text described in Appendix 1. The item 'expected accuracy' is the product of the count and its corresponding correct rate. From the table, the codes generated by the modified Huffman coding algorithm have the lowest hit count sum, which is almost 2/3 and 1/2 of those of 3 digits and 4 digits; meanwhile, its average expected accuracy of text input is comparable to the other 3 codes. This is realised by assigning the motion patterns with higher correct rate to more frequently used figures, which is guaranteed by the correct rate sorting

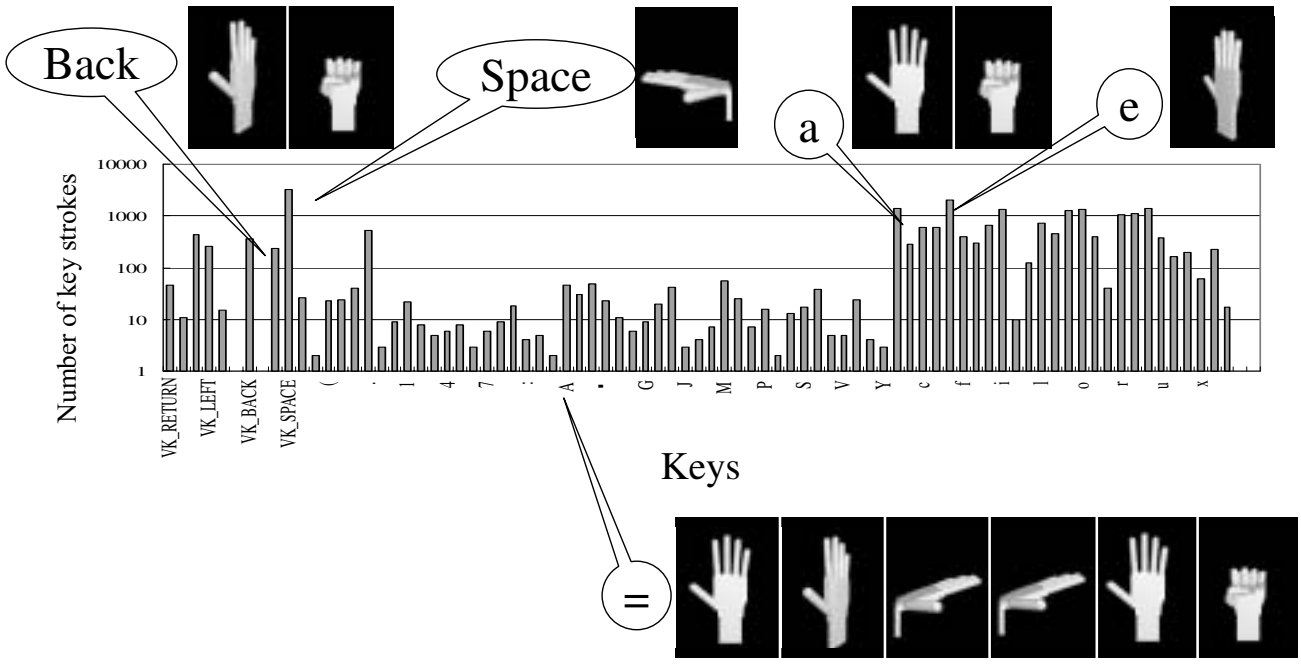

Figure 10 Keystroke occurrences and the Huffman code tree.

operation. This was reflected by the fact that the hit count values of the modified Huffman coding were in descending order – same as that of correct rates.

Text input experiment

The actual learning and training process is as follows:

1. *Gesture output training stage*. Several online learning training sessions were carried out until a high gesture recognition accuracy rate was reached. There was one training session per day. The term of the stage is individual dependent. Our experience is that most subjects can go through this stage in 3–5 days. Although the online learning method itself can deal with the time-varying characteristics, it is impractical to change a codebook after it has been used; the training for the online learning-based EMG-to-gesture classifier would last until comparatively stable action patterns could be generated.
2. *Codebook generation stage*. In the case of non-amputees, a group were asked to supply a piece of text specific to their particular field or occupation. They were then required to input the text at their normal input speed. Both the keystroke occurrence features and the individual characteristics in inputting were saved in a key code record. In the case of amputees, several existing key code records were employed in order to decide a suitable one. The key code records were analysed and used to generate codebooks.
3. *A text input exercise stage*. Subjects were asked to remember the codebook generated and practise text input.

Table 2 Motion number needed and expected average accuracy for the example text in Appendix 1

		Huffman code variable		*6 motions 3 digits*		*5 motions 3 digits*		*3 motions 4 digits*	
Motion code	*Correct rate*	*Hit count*	*Expected accuracy*	*Hit count*	*Expected accuracy*	*Hit count*	*Expected accuracy*	*Hit count*	*Expected accuracy*
Flexion(5)	0.85	**11 657**	9908.45	13 456	11 437.6	10 468	8897.8	34 124	29 005.4
Extension(4)	0.83	**8501**	7055.83	20 015	16 612.45	11 308	9385.64	21 929	18 201.07
Grasp(3)	0.82	**6372**	5225.04	9750	7995	21 995	18 035.9	32 887	26 967.34
Supination(6)	0.74	**5636**	4170.64	12 617	9336.58	10 551	7807.74	0	0
Open(2)	0.72	**4884**	3516.48	4101	2952.72	12 383	8915.76	0	0
Pronation(7)	0.72	**4490**	3232.8	6766	4871.52	0	0	0	0
Sum		**41 540**		66 705		66 705		88 940	
Average			**0.797045**		0.797629		0.795185		0.833976

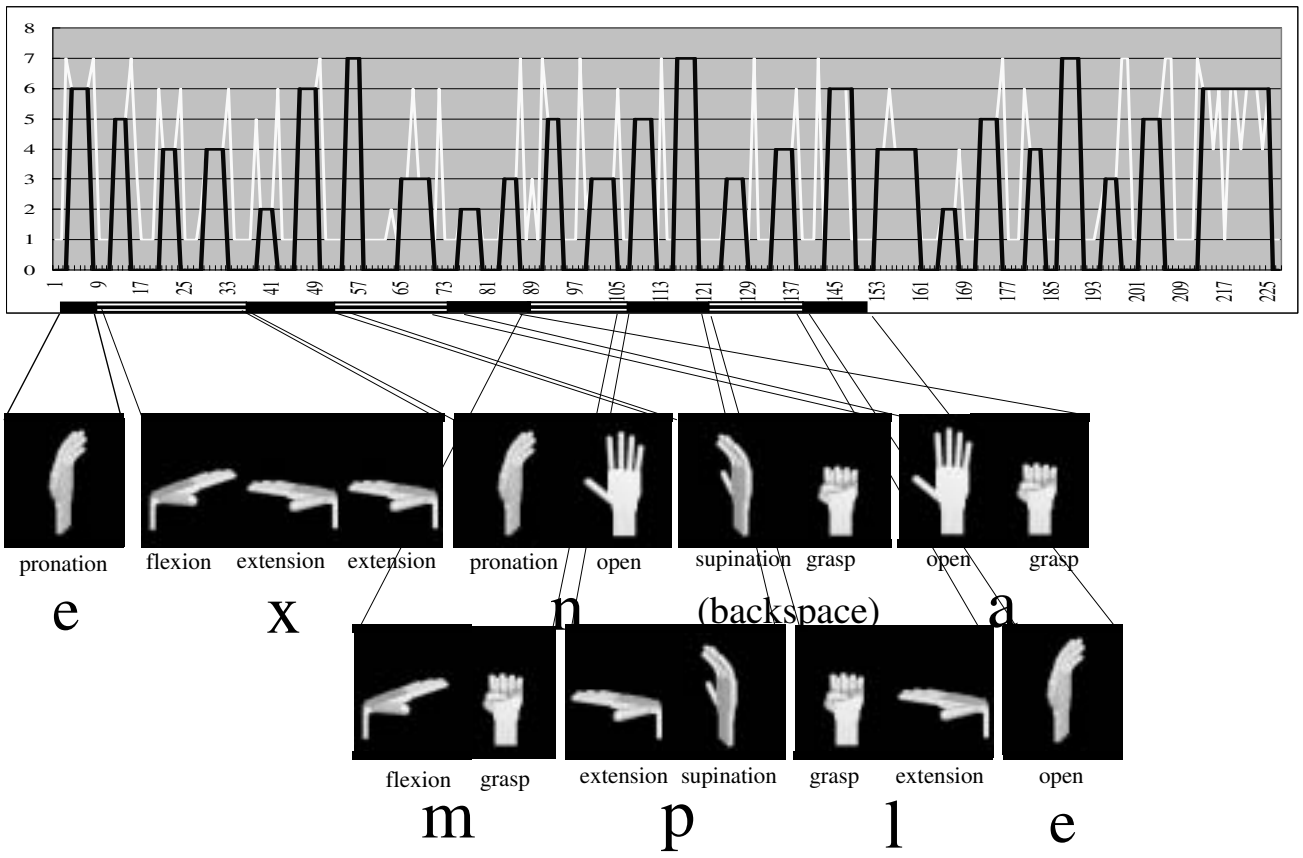

Figure 11 Text input experiment.

To test the overall performance of the integrated system, a text input experiment was done. For initial training, a short sentence, 'An important example of an information source is English text.', was used. The sentence has 10 words, 53 symbols (21 different symbols, including SPACE) and, if no mis-recognition or mis-input occurs, needs 96 motions to finish the input (by the codebook shown in Appendix 2). The average motion number for each symbol is 1.88.

Figure 11 shows a real motion sequence for inputting one word, 'example'. In the graph of Figure 11, the raw recognition results are in white, and the smoothed results are in black. The stop bits in smoothed results were shifted to '0' from '1' to achieve visual clarity. The vertical and horizontal axes are time and motion pattern code, respectively. The bars under the horizontal axis denote the motion compound to express one symbol, illustrated by corresponding CG hand icons. An input error occurred at the third character input, so that it became 'exn', whereas it should have been 'exa'. To correct the error, a 'backspace' was carried out, then, 'ample' was input. A non-amputee took part in a repeated text input experiment, using the codebook described in Appendix 2. Ten trials were done, and the motion number used for 10 trials is shown in Figure 12. Note that the difference between the value and 96 (the number of motions needed to input the text without any mis-operation) is the number used to input a 'backspace' to delete an incorrect symbol and input the correct one. During 10 trials, there was no clear change to the input accuracy and efficiency. The time to input the sentence is roughly 3–5 minutes. This is far from ideal even compared with Morse code. One reason for the length of time taken was that the subject was not sufficiently accustomed to the codebook. Another reason is that the motions used have quite a large range of movement (ROM), so the time for the motions is much longer than that of finger motions.

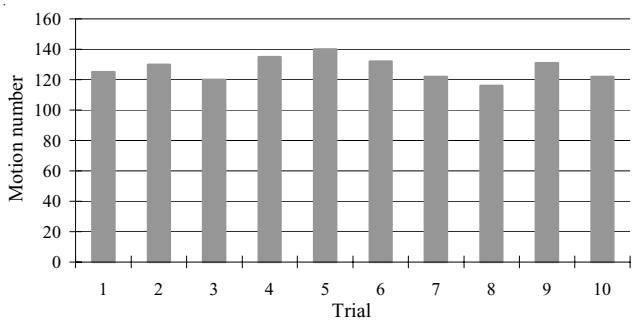

Figure 12 Motion number needed for text inputting.

However, the motions used in this experiment are those that can be used by forearm amputees.

A database that contains a variety of sentences and paragraphs, which can be selected according to individual needs and training situations to give operators a gradual improvement, is under construction. The results of database-based training will be described in a coming report.

Conclusions

To develop an EMG keyboard system that can be used by forearm amputees, the following were employed:

- an online learning method to recognise forearm gestures;
- a smoothing algorithm to extract motion labels from discrete gesture symbol sequences for continuous motions; and
- a modified Huffman coding algorithm for motion sequence-to-character transformation.

Results showed the possibility of realising large DOF input with the EMG keyboard, which can be expected to be a quiet and free-posture input means with compact (compared with speech recognition) support software and hardware.

Planned work includes:

1. A long-term observation experiment to identify the learning effect of EMG keyboard use, as well as posture and other ergonomic factors influencing input accuracy and efficiency.
2. Exploration of the possibility of using coding algorithms with redundancy so as to realise self-correction; moreover, smoothing algorithms taking a motion as a real continuous process will be considered.
3. Exploration of other muscle groups as control signal sources, especially muscle groups that are speedy and have small energy consumption, as in finger movement, teeth clicking etc.

Appendix 1

Keystroke occurrence frequency of the Preface (p xi–xviii) to the book *Bioinformatics – The Machine Learning Approach* edited by Pierre Baldi and Soren Brunak. The text includes 35 paragraphs, 3120 words and 19 035 characters (22 235 characters if SPACE included).

VK_RETURN	46	:	3	Y	2
VK_UP	10	;	4	a	1371
VK_RIGHT	426	=	1	b	291
VK_LEFT	261	A	46	c	613
VK_DOWN	14	B	30	d	601
VK_DELETE	0	C	47	e	2033
VK_BACK	368	D	22	f	393
VK_TAB	0	E	10	g	301
VK_COMMA	239	F	5	h	662
VK_SPACE	3200	G	8	I	1322
“	26	H	19	j	9
‘	1	I	42	k	123
(	22	J	2	m	451
)	23	K	3	l	734
-	40	L	6	n	1252
.	519	M	56	o	1308
/	2	N	24	p	389
0	8	O	6	q	39
1	21	P	15	r	1080
2	7	Q	1	s	1091
3	4	R	12	t	1392
4	5	S	16	u	386
5	7	T	37	v	164
6	2	U	4	w	200
7	5	V	4	x	60
8	8	W	23	y	222
9	17	X	3	z	16

Appendix 2

Huffman codebook generated (2 – open, 3 – grasp, 4 – extension, 5 – flexion, 6 – supination, 7 – pronation).

VK_RETURN	774	:	63465	Y	63466
VK_UP	6324	;	57663	a	54
VK_RIGHT	66	=	576654	b	76
VK_LEFT	72	A	775	c	42
VK_DOWN	6342	B	5764	d	47
VK_DELETE	57662	C	637	e	2
VK_BACK	74	D	5773	f	62
VK_TAB	57667	E	6325	g	73
VK_COMMA	575	F	57664	h	46
VK_SPACE	3	G	6322	I	53
“	5763	H	5772	j	6323
‘	576653	I	773	k	635
(	5774	J	63467	l	43
)	5767	K	63463	m	64
-	776	L	57656	n	52
.	65	M	636	o	56
/	576655	N	5762	p	67
0	6327	O	57653	q	772
1	5776	P	6343	r	44
2	57654	Q	576656	s	45
3	57667	R	6347	t	55
4	57657	S	6345	u	75
5	57655	T	777	v	572
6	63462	U	57662	w	573
7	57652	V	57666	x	633
8	6326	W	5775	y	574
9	5777	X	63464	z	6344

References

Aihara K, Douya K. 1993. Neuron, fuzzy, chaos – basic analog computing. Reading, MA: OHMSHA Publ.

Barreto AB, Scargle S, Adjouadi M. 1999. Real-time digital EMG/EEG signal processing in a human–computer interface for users with severe motor disabilities. In: Online Proceedings of the International Conference on Signal Processing Applications & Technology (ICSPAT) [CD-ROM, paper nr 355]. 1999 Nov 1–4; Orlando, FL, USA. Accessed 21 Oct 2003. URL: http//www.icspat.com

Barreto AB, Scargle S, Adjouadi M. 2000. A practical EMG-based human–computer interface for users with motor disabilities. *J Rehabil Res Dev*, 37:53–63.

Birbaumer N, Ghanayim N, Hinterberger T et al. 1999. A spelling device for the paralyzed. *Nature*, 398:297–8.

Katoh R, Nishikawa D, Yu WW et al. 2002. Evaluation of biosignal processing methods for welfare assisting devices – evaluation of EMG information extraction processing using entropy. *J Robotics Mechatronics*, 14:573–80.

Lueder R, Grant C. 1997. Alternative keyboard designs [online]. Accessed 23 Sep 2003. URL: http://humanics-es.com/alt-kb.pdf

Martin BJ, Rempel DJ, Dennerlein J et al. 1996. EMG analysis of muscle load in keyboard work. Paper presented at the International Conference on Occupational Disorders of the Upper Extremities. 1996 Oct 24–25; Ann Arbor, MI, USA.

Nishikawa D, Yu WW, Yokoi H et al. 2002. On-line supervising mechanism for learning data in surface electromyogram motion classifiers. *Syst Comput Jpn*, 33(14):1–11.

Smith W, Cronin D. 1992. Research study – ergonomic test of the kinesis contoured keyboard [online]. Accessed 23 Sep 2003. URL: http://kinesis-ergo.com/lab-data.htm

Yu WW, Yamaguchi H, Yokoi H et al. 2002. EMG automatic switch for FES control for hemiplegics using artificial neural network. *Robotics Autonomous Syst*, 40:213–24.