A wireless internet interface for person with physical disability

Cheng-San Yang \textsuperscript{a}, Cheng-Huei Yang \textsuperscript{b}, Li-Yeh Chuang \textsuperscript{c}, Cheng-Hong Yang \textsuperscript{d,*}

\textsuperscript{a} Department of Physical Medicine and Rehabilitation, Chiayi Christian Hospital, Chiayi, Taiwan
\textsuperscript{b} Department of Electronic Communication Eng., National Kaohsiung Marine University, Kaohsiung, Taiwan
\textsuperscript{c} Department of Chemical Eng., I-Shou University, Kaohsiung, Taiwan
\textsuperscript{d} Department of Electronic Eng., National Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan

**Abstract**

Technologically assistive devices are increasingly playing more important roles in the lives of persons with disabilities, with one of the more promising considerations being a combination of the functions of computer software and hardware. However, using a conventional keyboard for Internet access is prohibitive for persons whose hand coordination and dexterity are impaired by ailments such as amyotrophic lateral sclerosis, multiple sclerosis, muscular dystrophy, and other severe handicaps. To assist participants with physical disabilities in sharing the resources of the Internet, we designed and implemented an easy-to-operate wireless input interface using Morse code as an adaptive communication tool. Moreover, an adaptive Morse code recognition process is introduced. After two months’ practice on this system, three participants with physical disabilities could conveniently gain access to the Internet.

1. Introduction

The World Wide Web (WWW) was introduced to the Internet in 1989, and since then the number of Internet users has increased exponentially every year. Likewise, an abundance of applications have emerged, which utilize the Internet to provide information services, entertainment, and communication services to users, thus making learning, working, and our daily lives more convenient and efficient than ever before. Unfortunately, an unmodified computer cannot be used to its potential by people with disabilities. They require customized additional adaptive tools and interfaces. Consequently, one trend in high technology production now is to develop adaptive tools for persons with disabilities in order to assist them in self-teaching, personal growth, and ensure their independence. Among the various technological adaptive tools available, many are based on the adaptation of computer hardware and software. The areas of application for computers and these tools include training, teaching, learning, rehabilitation, communication, and adaptive design \cite{1-4}.

Many computer assisted key-in systems, such as a head mouse, a mini-keyboard, a king-keyboard, trackball, joystick, alternative keyboard, and keyguard have been developed specifically for the use of persons with disabilities. However, a person whose hand coordination and dexterity is impaired by ailments such as amyotrophic lateral sclerosis, multiple sclerosis, muscular dystrophy, etc. cannot share the resources on the Internet due to the lack of a suitable adaptive communication device and computer input interface. Therefore, in this study, Morse code, with a wireless single-switch input system, was selected as a communication adaptive device for the persons with disabilities. Morse code has been shown to be an excellent candidate for a communication adaptive device \cite{5-13}. An easily operated interface was provided and an

\* Corresponding author.  
E-mail address: chyang@cc.kuas.edu.tw (C.-H. Yang).
elicit recognition method using support vector machines was developed and implemented. By using this communication interface system, persons with disabilities were able to access Internet resources easily. Experimental results revealed that three participants with physical disabilities were able to gain access to resources on the Internet after two months’ practice with the new system.

2. System design

Morse code is a simple, fast, and low-cost communication method composed of a series of dots, dashes, and intervals in which each character entered can be translated into a predefined sequence of dots and dashes (the elements of Morse code). A dot is represented as a period “.”, while a dash is represented as a hyphen, or minus sign, “-“. Each element, dot or dash, is transmitted by sending a signal for a standard length of time. Based on the definition of Morse code, the tone ratio for dot to dash must be 1:3. This means that if the duration of a dot is taken to be one time unit, then that of a dash must be three time units. In addition, the silent ratio for dot–dash space to character-space also has to be 1:3. In other words, the space between the elements of one character is one unit while the space between characters is three units [9].

A block diagram of the Internet access system is shown in Fig. 1. When a user presses the single-switch Morse code input device, the signal is detected by the key scan circuit, and then the key data will be transmitted to computer through the RF circuit. All data will be transmitted to the Morse code control module for further processing. A circuit diagram of the Morse code input device is shown in Fig. 2. An 8051 single chip has been adopted to handle the communication between the input switch and the personal computer. Even though it only has small capacities of memory and I/O compared to a typical PC, it is still powerful enough to control the device. The 8051 internal serial communication function is used for data transmission and reception. To achieve the data communication at both ends, the TxD and RxD pins are connected to the TxD and RxD pins of an RS-232 connector. Then the two pins are connected to the RxD and TxD of an UART (Universal Asynchronous Receiver Transmitter) controller on the PC. The data communication protocol adopted is asynchronous transmission, with a 9600 bps baud rate, 8 data bits, one stop bit, and a nonparity check [14]. A single-switch Morse code entry key is used, and an audio side tone is provided for feedback. The entry switch consists of a large press-button, which can easily be handled by users with limited hand coordination.

An initialization command was added to the hardware circuit. Only after having received this initialization command sent from the computer the circuit will start to process the key-press operation. The circuit will then reset the device by clearing all data regarding the key-press status and the data register and a signal is sent to the PC end to indicate that the device is ready. Afterwards, all operations of the Morse code input switch will be monitored. When a key is triggered (pressed or released), a program in the 8051 single chip will detect which key was triggered. Then, the corresponding code and the key-press or key-release data combination will form a byte data, which will be sent to the data register. The coded data is sent out in the RF circuit of the Tx block. When the Rx block receives the data, the 8051 chip decodes the data and send it to the data register. When data arrives in the data register via the serial port, the transmission function will be enabled. As the serial port begins to transmit data, the parallel data in the data register will be transformed into serial data. The serial data will be sent to the RxD pin of the PC server through a TxD signal line. After these signals are received by the UART controller.
Fig. 2. Circuit diagram of the Morse code input device.

of the PC, the UART will decode them into character data, save them in a data register, and then inform the CPU that there are incoming data. The data will then be transmitted back to the hardware which indicates that the data has been received and that the next data can now be sent. Now, the circuit will begin to process the next key-press or key-release operation.

The interface can be connected to a PC via a personal computer, a standard RS-232 serial port which has two advantages:

1. Installation is completed by simple plugging the device into the serial port of a PC.
2. Since the serial port is a standard communication device in the Windows operating system, installation of additional drivers and rebooting is not necessary.

Internet applications are increasingly popular and information can be retrieved quickly. For persons with disabilities however, services are often inaccessible, since they require proper adaptive tools and interfaces to use ordinary computers. This can be achieved with a Morse code recognition module.

2.1. Interface control module

Several keys have been edited to allow users easy Internet access. Two standard input devices are accepted by the system namely: the mouse and the keyboard. The software discussed in this study is similar to a standard mouse driver. The difference is that a single-switch Morse code input system replaces the mouse. Therefore, a user’s input has to be converted first, and then the converted results will be sent to the operating system and distributed to each application program. The loaded Morse code recognition module monitors all input from the keyboard. The system records the time length of a key-press and a key-release when a user keys in data, and then calculates the time intervals of every key-press and key-release. Then, the data is sent to a Morse code recognition module to determine the length category of a key-press or a key-release signal, i.e. a dot or dash, or an interval between a dot, dash, and space. The length category is compared to a Morse code reference table, which locates the corresponding character. This character is then sent to the interface control module and entered into the keyboard data stream. A corresponding message is sent to the application program that is currently being executed.

Fig. 3 shows the graphical interface of the Morse code input system, including the Morse code for commonly used keys, such as numbers and letters, as well as for some special keys. In the table shown in Fig. 3, the characters are selected by first entering the Morse code shown in a row, followed by the Morse code of the column. Thus, to choose the letter “a” for example, the Morse code “--.” is entered.

The input interface (Fig. 3) has the following characteristics:

1. The Morse code input switch allows users to switch between standard keyboard input and Morse code input.
2. Functions setup: users can set details of input conditions based on personal requirements and preferences.
3. Morse code reference table: the provided Morse code reference table can be shown in an open window so users can key in the desired code.
2.2. Morse code recognition module

The proposed method is divided into five modules: tone recognition, space recognition, learning process, adaptive processing, and character recognition. A block diagram of the Morse code recognition process is shown in Fig. 4. Initially, the input data stream is sent individually to either the tone recognition module or the space recognition module, depending on switch-down time (tone element) or switch-up time (space element). In the tone recognition module, the tone element value is recognized as either a dot or a dash, and then sent into the learning process (support vector machines, SVMs). Simultaneously, the recognized tone element (dot or dash) and each successive tone element are saved in a dot–dash buffer and a tone element buffer in the tone buffer section. The space element value is recognized as being either a dot–dash space (the space between elements of one character) or a character space (the space between characters) in the space recognition module, and then fed directly into the adaptive processing module. Once a character space is obtained, the value(s) in the tone buffer is (are) sent to the character recognition module, which identifies this character.

A Morse code character $x_i$ can be represented by:

$$e_1(x_i), b_1(x_i), \ldots, e_j(x_i), b_j(x_i), \ldots, e_n(x_i), b_n(x_i) \quad 1 \leq j \leq n$$

where
\( e_j(x_i) \): when a key is pressed down, it is presented as either a ‘dot’ or ‘dash’, the duration of the \( j \)th Morse code element of the input character \( x_i \).

\( b_j(x_i) \): when a key is held up, it is presented as one of two spaces: dot–dash space or character space, the duration of the \( j \)th space of the input character \( x_i \).

\( m_j(x_i) \): a dot or dash recognized from \( e_j(x_i) \).

\( n \): the total number of Morse code elements in character \( x_i \).

**Tone recognition**

Initially, each tone_element is normalized to obtain an input value within a range of \(-1\) to \(1\).

\[
\tilde{x} = 2.0 \ast \frac{(\text{tone_element} - 0.5 \ast (\text{tone}\_\text{max} + \text{tone}\_\text{min}))}{\text{tone}\_\text{max} - \text{tone}\_\text{min}} \tag{1}
\]

where \( \text{tone}\_\text{max} \) and \( \text{tone}\_\text{min} \) are the largest and smallest values of the tone element respectively. If a tone_element is larger than the \( \text{tone}\_\text{max} \) value, then the \( \text{tone}\_\text{max} \) value is substituted by this tone_element value. If a tone_element is smaller than the \( \text{tone}\_\text{min} \) value, then the \( \text{tone}\_\text{min} \) value is substituted by this tone_element value. The obtained value \( \tilde{x} \) can be sent into a decision function \( f(x) \) to recognize the value as being either a dash \( (f(x) \geq 0) \) or a dot \( (f(x) < 0) \). The decision function can be written as:

\[
f(x) = \text{sign} \left( \sum_{i \in I} \alpha_i y_i K(x_i, x_j) + \beta \right) \tag{2}
\]

where \( \alpha_i \) is the solution of the constrained maximization problem, \( y_i \in \{-1, +1\} \) and \( \beta \) is the bias \([16]\). The kernel function used is a radial basis function (RBF), such as the Gaussian function

\[
K(x_i, x_j) = \exp \left( -\frac{\|x_i - x_j\|^2}{2\sigma^2} \right), \quad i = j = 1, 2, \ldots, l. \tag{3}
\]

The new tone value of the input stream will be entered into the decision function \( f(x) \) to determine the value as being either a dash or a dot. Once the tone value is identified, it can be labeled and sent into the training data set. Then the training process is performed to recalculate the decision function.

At the beginning of this process the initial \( \text{tone}\_\text{base} \) (TB), which is used to serve as the initial dot–dash classifier, is absent. To determine the initial TB, the first nine values of tone elements are taken as reference values and sorted in descending order. Once the sorting is complete, the element values are compared to each other in order to determine their relationship, and are recognized as either “dash” or “dot”. “Dash” means that a value is at least twice as large as any other value. A smaller value is defined as a “dot”. After the “dash” or “dot” relationship is determined, the \( \text{dash}\_\text{base} \) and \( \text{dot}\_\text{base} \) represent the average of the dash values and the dot values. The resulting final values represent the initial TB.

\[
\begin{align*}
\text{tone}\_\text{sum} &= \text{dash}\_\text{base} + \text{dot}\_\text{base} \\
\text{tone}\_\text{ratio} &= \text{dash}\_\text{base}/\text{dot}\_\text{base} \\
\text{TB} &= \text{tone}\_\text{sum}/\text{tone}\_\text{ratio}.
\end{align*}
\]

Once the initial training data set is determined, it can be used in the learning procedure, and the initial decision function for the dot–dash classifier can be determined.

**Support vector machines**

Support Vector Machines (SVMs) are based on the theoretical learning theory developed by Vapnik \([15,16]\). Support vector machines (SVM) have proven to be highly effective for a number of real world problems, including recognition of handwritten digits, 3-D objects, breast cancer prognosis, and engine-knock detection \([17–20]\). They demonstrate an impressive resistance to overfitting in classification and their training is performed by maximizing a convex functional, which means that there is a unique solution that can always be found in polynomial time. The original input space is mapped to a high-dimensional dot product space called feature space, and an optimal hyperplane is determined to maximize the generalization ability.

In this study, SVM algorithms are applied to dots or dashes of Morse code recognition. However, training this system is nontrivial and a high cost of computation is required by the use of optimization packages. Kernel–Adatron (KA) algorithms \([20,21]\) are used to emulate SVM training procedures, but adapted by the introduction of kernels so that they can find nonlinear decision boundaries in the high-dimensional feature space. This is a fast and simple learning procedure, which finds a maximum margin hyperplane in a high feature space. Experimental results have shown that the predictive power is equivalent to that of an SVM and the running time can be orders of magnitude faster \([20]\). The KA procedure \((\eta = 0.1)\) is shown below:

1. Initialize \( \alpha_i^0 = 0 \).
2. For \( i = 1, \ldots, m \) execute step 3, 4 below.
(3) For a labeled point \((x_i, y_i)\) calculate:
\[
z_i = \sum_{j=1}^{m} \alpha_j y_j K(x_i, x_j).
\]

(4) Calculate \(\delta \alpha_i^t = \eta (1 - z_i y_i)\):

\((4.1)\) If \((\alpha_i^t + \delta \alpha_i^t) \leq 0\) then \(\alpha_i^t = 0\).
\((4.2)\) If \((\alpha_i^t + \delta \alpha_i^t) > 0\) then \(\alpha_i^t = (\alpha_i^t + \delta \alpha_i^t)\).

(5) If a maximum number of iterations is exceeded or the margin \(\lambda\) is approximately 1 then stop, otherwise return to step 2.

\[
\lambda = \frac{1}{2} \left[ \min_{|y_i| = +1} (z_i) - \max_{|y_i| = -1} (z_i) \right].
\]

**Space recognition**

The space recognition module detects the spaces existing between entire characters, as well as the space between the isolated Morse code elements that comprise a unique character. When the data stream of characters composed of Morse code elements is entered, these elements must then be identified as being either a dot–dash space (space between entire characters) or a character space (space between isolated elements of a character). The procedure for this character detection is shown below:

1. initiate \(j = 1\).
2. if \(b_j(x_i) < \text{silence}_\text{base}\), then go to step 3, otherwise go to step 4.
3. \(b_j(x_i)\) is a dot–dash space. Let \(j = j + 1\) and go to step 2.
4. \(b_j(x_i)\) is a character space. Then a sequence of tone durations between the character spaces is obtained. Go to step 1.

Initially, the first character \(x_i\) cannot be immediately isolated due to the initial silence_base (SB) value being absent. Subsequently, the initial SB is obtained by extracting the first nine values of silent elements entered as reference values; afterwards, all values taken are arranged in descending order, and the relationship among each value is then compared. If a value is found to be twice larger than any other value, this value is designated as being long (L), and all smaller values are designated as being short (S). Once all relationships have been established, the average of the nine references values can be calculated and assigned to be the initial SB.

After the initial SB value has been determined, it can be sent into the adaptive processing module as the initial value. Meanwhile, the character detection equation can be used to calculate a subsequent SB value based on this obtained SB value to recognize spaces within elements. After a space element has been recognized, the SB value can be recalculated. If the result shows L, the space element is divided by 3.0, and the obtained value is only then sent into the adaptive processing; otherwise, the space element is directly sent into the adaptive processing module to obtain a new SB. Whenever an SB is obtained, the data stream is separated into elements and spaces. After the Morse code elements of a character have been isolated from a data stream, the elements can be recognized in the character recognition module [21].

**Adaptive processing**

The variable degree variable step least–means-square (VDVSLMS) algorithm used here serves to change the standard ‘space’ length [23]. The average of space \(b_i(x_i)\) \((i = 1, n - 1)\) in \(x_i\) is the \(i\)th input data of the algorithm. The VDVSLMS algorithm utilizes the current data to compute a new weight vector using the weight update recursion of the standard LMS algorithm with step size \(\mu\) [24]. The new weight vector together with the current data are then utilized to again update the desired weight vector using the standard LMS algorithm weight update recursion with step size \(\mu\). Each adaptive weight, \(W(n)\), is adjusted according to the equation

\[
W(n + 1) = W(n) - \alpha_2(n) \hat{\nabla}(n)
\]

where

\[
\alpha_2(n) = 2\mu (1 - \mu X^T(n)X(n)).
\]

The subscript on \(\alpha(n)\) is used to indicate the degree, and

\[
\hat{\nabla}(n) = -2\varepsilon(n) X(n)
\]

is an estimate of the gradient.

\[
\varepsilon(n) = d(n) - X^T(n)W(n)
\]

where \(d(n)\) is the scalar desired signal. \(\mu\) is the step-size parameter that controls the speed of convergence as well as the steady-state and/or tracking behavior of the adaptive filter. The step size \(\mu\) has a value of 0.02 in our system [21].
Character recognition

Once a character space value has arrived in the tone_buffer, the elements in the tone buffer have to be sent to the character recognition process. If the recognized character set can be directly matched to a code set from the Morse code table, then it is immediately translated using the Morse code table. Otherwise, it has to be translated by the following minimum distance calculation. First, each tone element value in an unknown tone element stream is divided by the tone_base of the previous tone element set. Then, the distances between each tone value and the code elements in each character of the Morse code table are calculated. The character with the minimum Euclidean distance to the tone value is chosen as the value for the unknown character. The procedure for the shortest Euclidean distance method is the following. First, each tone element, \( e_j(x_i) \), is divided by the tone_base. Then, the roots of the sum of the square distances between the new tone element and the character in the Morse code table are calculated. The character in the Morse code table that has the shortest Euclidean distance is recognized as the unknown character [21].

3. Results and discussion

Based on accounts from users with disabilities, maintaining a stable typing speed is an insurmountable challenge. Interestingly enough though, a Morse code time series is also an unstable entity, unstable in speed and/or in rate. Maintaining precise intervals is a difficult task even for persons without disabilities. Subsequently, an unstable typing speed or rate may generate two types of errors: space recognition errors and tone recognition errors. Generally speaking, a person's typing rate is constant over a short period of time, meaning that a person's typing rate at a given time is similar to the typing rate of the immediately preceding several words. Therefore, in this study, the criterion to distinguish “dot or dash” or “dot–dash space or character space” was recalculated when the Morse code element was generated, helping to increase the recognition rate. Training a user to input code using the proposed system does not differ from standard Morse code input training.

The system can be installed under Windows XP and Windows Vista environments. Figs. 5a and 5b below illustrate how data is entered into the user interface, i.e. a commonly used browser. It is also applicable in many other standard programs used by a computer user, e.g. email and word processing programs, etc.

Three persons with disabilities were chosen as test participants to investigate the efficiency of the proposed system. They practiced on this system for two months. Participant 1 (P1) was a 14-year-old male adolescent who has been diagnosed with cerebral palsy. His voluntary movements were accessible, but an initial delay was evident before the movement was initiated. The involuntary movement partially disrupted the volitional movement, making it uncoordinated. An IQ (Intelligent Quotient) test showed his intelligence to be normal. Although his hearing and cognition abilities were normal, he exhibited marked speech difficulties. Participant 2 (P2) was a 14-year-old female adolescent with cerebral palsy, athetoid type, who experiences involuntary movements of all her limbs. Her IQ is relatively high, but dysarthria is noted, resulting in difficulty of verbal communication. Participant 3 (P3) was a 40-year-old male adult, with a spinal cord injury and incomplete quadriplegia due to an accident. His right wrist is limited in its functions and his individual finger movement is also limited, which results in dysfunctional hand movement. His intellect and ability to verbally communicate are not impaired. Initially, the participants were unfamiliar with the operation of the proposed system. Each person typed five selected uniform resource locators in a test for a total of five times during a two-month period [22]. After repetitive training sessions the
typing speed of the above users progressively improved. Typing speed improved by about 11%. According to the training results, the ratio variation coefficient was larger than the speed variation coefficient. This means that it was more difficult to maintain a stable ratio than a stable speed. After two months' practice, the speed variation coefficient and ratio variation coefficient had improved from 52 down to 31 and from 32 down to 24, respectively.

4. Conclusions

We designed and implemented a wireless human interface for Internet access under a Windows environment for persons whose hand coordination and dexterity are impaired by severe handicaps. Morse code was selected as the adaptive communication device, introducing a method with a high recognition rate for this code. A reliable Morse code recognition method is important for persons with disabilities, helping to compensate for an unstable input rate, and correcting user errors resulting from long periods at the computer. The system provides an easy-to-operate environment and allows a user with a disability to obtain information easily from Internet resources. Experimental results revealed that three participants were easily able to gain access to resources on the Internet after two months' practice with the new system.

Acknowledgements

This work was supported by the National Science Council, R.O.C., under contract NSC 92-2218-E-151-001, 93-2213-E-151-014, 94-2614-E-151-001, and 95-2221-E-151-004-MY3.

References