Two-bit quartering, a new input mechanism based on Morse code.

This is a research report on two-bit quartering, a new technique to communicate with just two switches. It is intended for readers with a background in AAC and/or in psychology and an interest in communication techniques. Most of it is now better treated in twobitq.pdf.

Comments to the author j.verrips@planet.nl

Finished May, 2003, minor changes July 2004, June 2005

Thanks to John Todman, Dundee, UK and Horabail Venkatagiri, Ames, USA. Both offered valuable advice and moral support.

Introduction

Some people need electronic switches to communicate. Different types of switches react to movement, to eye-blink, to air-flow (a sip-and-puff switch), or even to isometrical muscle contraction. Most switch-users can neither write nor type nor speak. If they are able to spell, they may use a projected keyboard and so called row-column scanning to enter text in a computer, where it may be spoken. Scanning is easy to learn but slow and tiring to execute. An alternative to scanning is Morse code, harder to learn but significantly faster. Two-bit quartering is a new switch-based technique that combines scanning and Morse code. These techniques are analysed in detail, switch counts are determined with a simulated experiment, speed of learning is studied with several able-bodied test subjects and an application is described intended for patients with Amyotrofic Lateral Sclerosis.

Input mechanisms with switches

In classical row-column scanning a focus moves by itself, pauses on every field and displays as a highlighted area. Three clicks serve first to start, then to select the correct row and next to select the correct column. Users must watch this focus, cannot look elsewhere and have a rather passive role. In Figure 1, the focus starts at Ø. In Figure 2 it has moved three times downwards and twice to the right. Figure 3 shows the number of pauses per field.

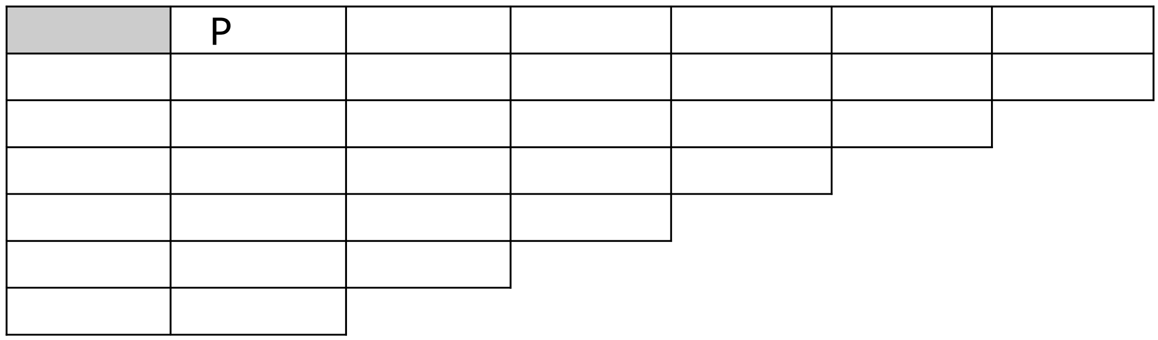


Figure 1. Triangular scanning matrix with empty first column.

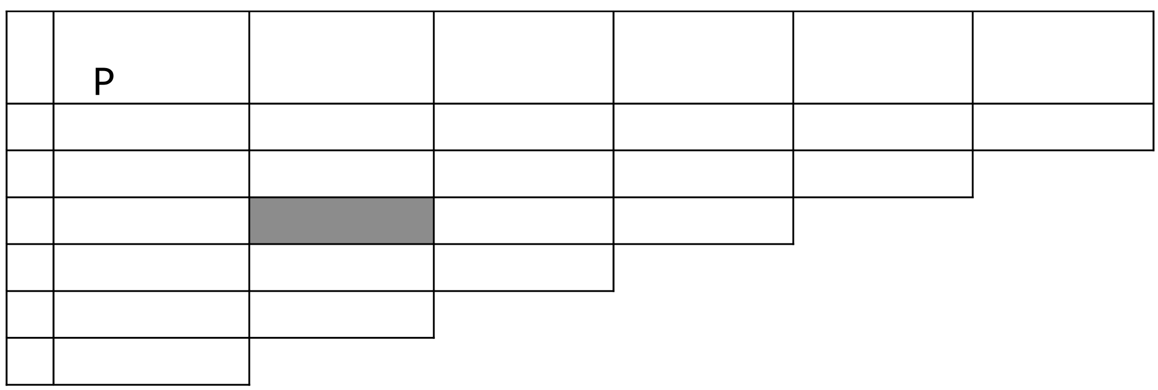


Figure 2. Figure 1 after three movements downwards and two to the right. If ‘d’ is now selected, the scan matrix returns to its’ state in Figure 1.

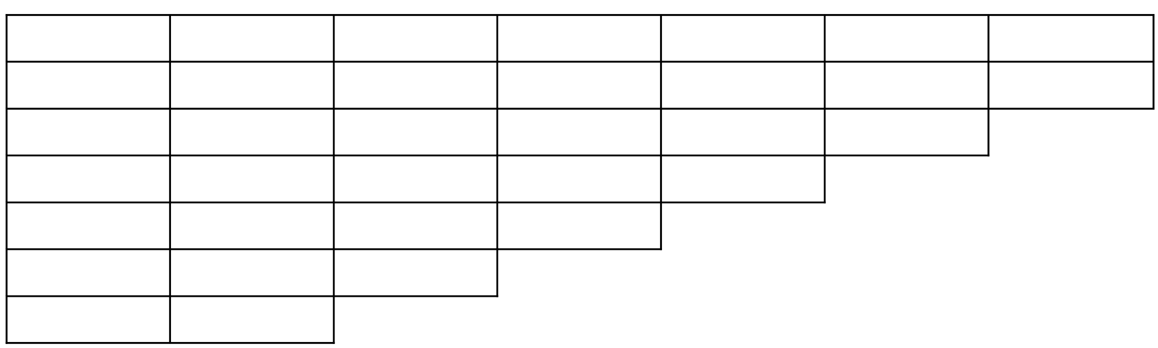


Figure 3. Same matrix as in Figure 1 and number of pauses per field with row-column scanning.

A related mechanism is converse scanning. Clicks move the focus and a pause selects the letter in the current field. With two switches one switch is used for rows and the other one for columns but converse scanning is also possible with one switch and two pauses. An empty first column allows to first select the row, wait a variable pause then select the column. The empty field 'obliges' the focus to wait, later the focus times out on a pause, selects the letter and starts again.

The number of clicks per character with converse scanning is equal to the number of pauses per character with classical passive row-column scanning. Of course, the ordering of the matrix determines this 'distance' for every character. For simulation studies about different orderings of rc-matrices consult Venkatagiri 1999. Other interesting articles on scanning are VanderHeiden, 1988, and Damper, 1984. Lesher et al, 1998, as well as Simpson and Koester, 1999, studied combined scanning and text-prediction.

Another input method with switches is Morse code, see Figure 4. Morse can be entered either with one, two or three switches. In this paper – means one switch and . the other. Codes for M –– or K –.– and SP ..–– or BS –––– may be exchanged as M and K are not frequent and have short codes. Computer access by Morse code is available and includes mouse functions, see Lynds, 2001. Once learned it is faster then row-column scanning and becomes fully automated. Different authors disagree about the time needed to learn Morse, due in part to individual differences. It is usually more than ten hours and, just as with ten finger blind typing, speed may continue to increase for several years. See King, 2000, and Standage, 1998, for more information on the subject.

A	**.-**	**F**	**..-.**	**K**	**-.-**	**P**	**.--**	**U**	**..-.**
B	**-...**	**G**	**--.**	**L**	**.-..**	**Q**	**--.-**	**V**	**...-**
C	**-.-.**	**H**	**....**	**M**	**--**	**R**	**.-.**	**W**	**.--**
D	**-..**	**I**	**..**	**N**	**--.**	**S**	**...**	**X**	**-..-**
E	**.**	**J**	**.---**	**O**	**---**	**T**	**-**	**Y**	**-.--**
SP	**..--**	**BS**	**----**	**Ret**	**.-.-**			**Z**	**--..**
1	**.----**	**2**	**..---**	**3**	**...--**	**4**	**....-**	**5**	**.....**
6	**-....**	**7**	**--...**	**8**	**---..**	**9**	**----.**	**0**	**-----**

Figure 4. Morse code, including numerals. Note that frequent letters have short codes.

Two-bit quartering uses codes to select groups of characters. In the scan matrix of Figure 5 four codes choose a quarter; . (the first switch) – (the second switch) : (twice the first switch) and = (twice the second switch). Figures 6 and 7 show how a second choice or sometimes a third choice allows to select individual characters. Note that . or – means a switch followed by a pause. Two more codes are used: . immediately followed by – is called +, and – immediately followed (no pause) by . is called ?. + selects a frequently used character such as SP or Space. =+ may select BS or BackSpace and –+ may select Ret or Return. Also, the shortcut may serve as a repeat function for arrows and function keys. They are used often and are displayed just below Figures 5 to 7.

. –

<table>
<tr><td>a</td><td>e</td><td colspan="2">n</td><td colspan="2">t</td></tr>
<tr><td>i</td><td>o</td><td colspan="2">r</td><td colspan="2">s</td></tr>
<tr><td rowspan="2">d</td><td rowspan="2">h</td><td>b</td><td>c</td><td>j</td><td>k</td></tr>
<tr><td>f</td><td>g</td><td>p</td><td>q</td></tr>
<tr><td rowspan="2">l</td><td rowspan="2">m</td><td>u</td><td>v</td><td>y</td><td>Z</td></tr>
<tr><td>w</td><td>x</td><td>BS</td><td>,</td></tr>
</table>

: + SP **=**

Figure 5. Matrix for two-bit quartering with two switches. Note four codes in four corners that may select four different states. The shortcut + is now bound to SP.

. –

a	e	n	t

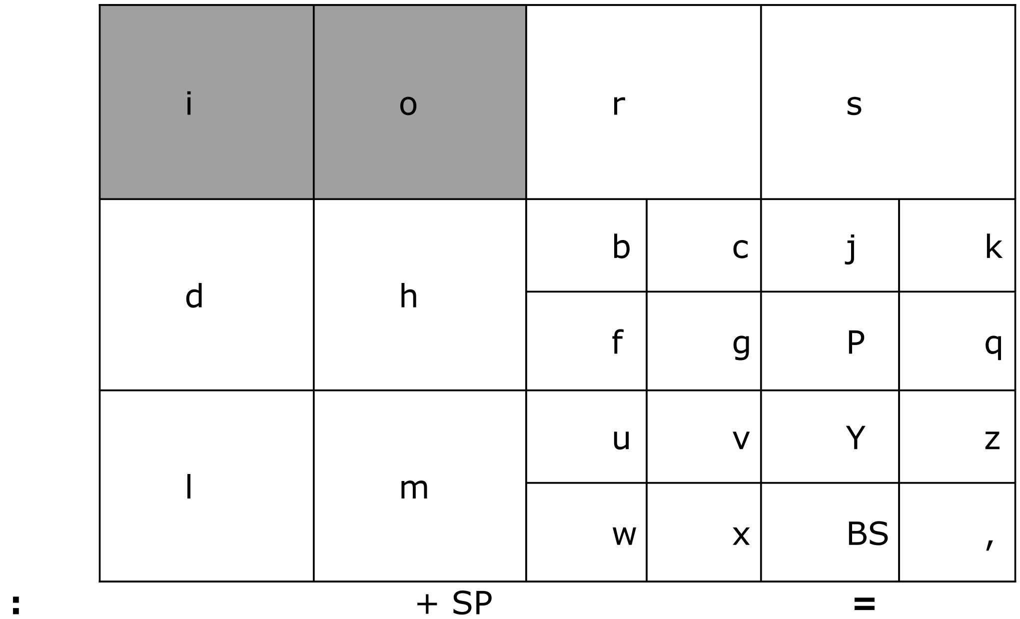
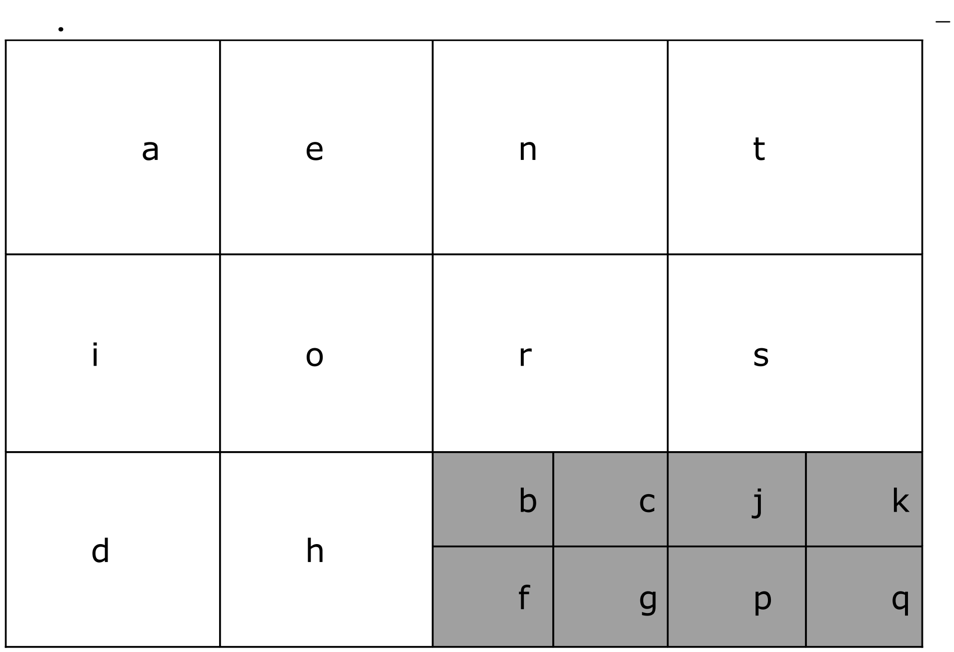


Figure 6. If in Figure 5 a user activates . the left upper quarter is marked on screen. Now . will select a, – will select e, : i and = o. If ? is entered to correct one sees Figure 5 again.



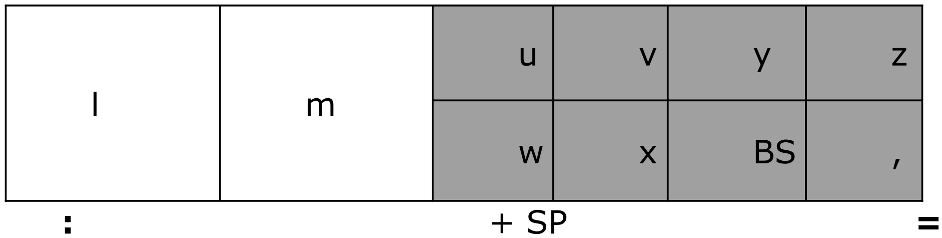


Figure 7.If in Figure 5 = is chosen users see its right lower quadrant marked. Readers may check that in Figure 5 =.– selects letter c. The shortcut + is now bound to BS.

? (the second switch followed by the first switch) serves to correct if the wrong quarter is chosen. If no quarter was chosen yet, as in Figure 5, ? shows the matrix of Figure 8 to choose from less frequent characters.

. –

A	B	E	F	Q	R	U	V
C	D	G	H	S	T	W	X
I	J	M	N	Y	Z	SP	BS
K	L	O	P	.	,	Ret	Tab
1	2	5	6	F1	F2	F5	F6
3	4	7	8	F3	F4	F7	F8
9	0	:	;	F9	F10	Alt	Ctrl
+	-	*	/	\	Esc	Caps	Shift

: + SP =

Figure 8. 4 x 4 x 4 = 64 different items can be selected with three choices out of four. In this example, after . . users may choose from A, B, C and D. Obviously, it might be filled with different characters, icons or commands.

Repeated quartering with variable depth has been described with a joy-stick, see VanderHeiden, 1988, but not with switches and Morse code, hence the addition 'two-bit'. It can be considered as a way to assign short codes to frequent characters and as a scanning method with diminishing group size and might be applied in very different contexts as to choose from pictures or icons without a mouse, or to select ideograms or phone numbers or even to construct sms messages with just two keys. Figure 9 offers a schematic overview, Figure 10 shows codes per character, Figure 11 shows a faster matrix.

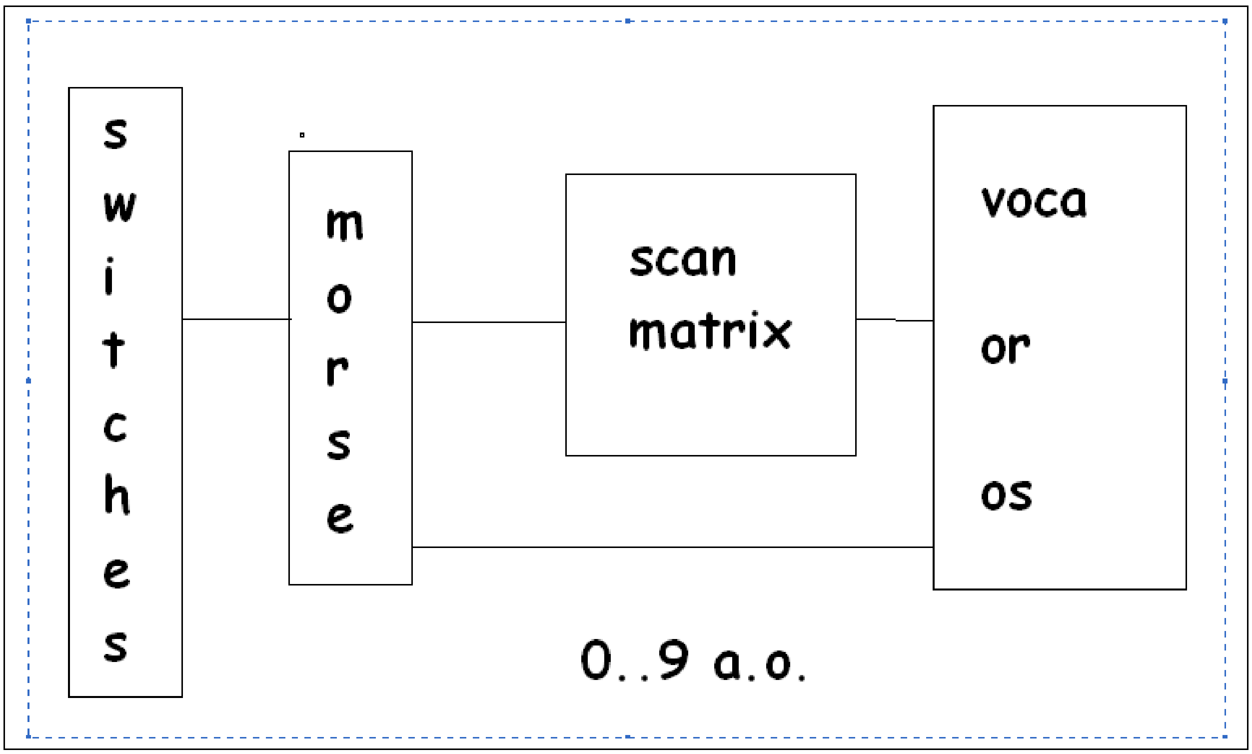


Figure 9.Schematic view of two-bit quartering. The second box on the left may consist of specialized hardware for Morse code or of switches wired to an opened keyboard and some extra software. If a code for a numeral 0..9 is entered the scan matrix passes it on to the operating system (os) and/or the voca, voice output communication aid, and the scanning mechanism is bypassed. In a sense the number of shortcuts is increased, at the price of learning more codes.

A	**..**	**F**	**=.:**	**K**	**=--**	**P**	**=-:**	**U**	**=:.**
B	**=..**	**G**	**=.=**	**L**	**::**	**Q**	**=-=**	**V**	**=:-**
C	**=.-.**	**H**	**:-**	**M**	**:=**	**R**	**-:**	**W**	**=::**
D	**:..**	**I**	**.:**	**N**	**-.**	**S**	**-=**	**X**	**=:=**
E	**.-**	**J**	**=-**	**O**	**.=**	**T**	**--**	**Y**	**==.**

SP	**+**	**BS**	**=+**	**Ret**	**-+**	**Z**	**==-**

Figure 10.Codes of two-bit quartering with Figure 5. These codes determine switches per character as well as pauses per character. Readers may verify that the number of pauses in the code of x (=:=) is three, or p_x=3 and that its number of switches is six, s_x=6. Users who can generate these codes from Figure 5 shall probably be able to operate two-bit quartering with eyes closed.

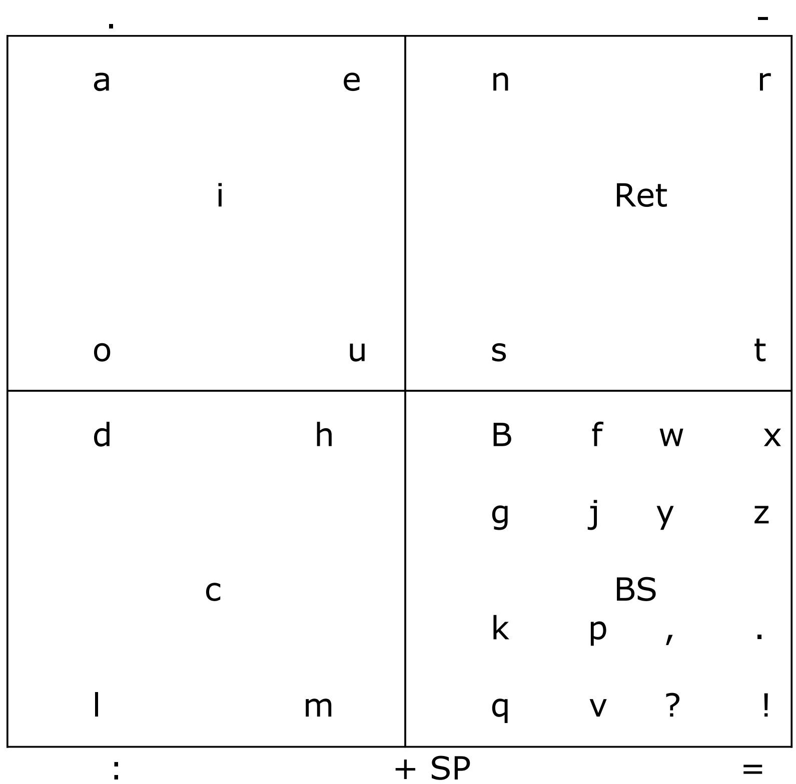


Figure 11. Alternative ordering. The value of the shortcut + is depicted in the middle of the different quarters. Vowels, frequent non-vowels, less frequent non-vowels and other characters all have their own square. Thus, =+ selects BS or BackSpace and :+ selects ‘c’.

Example

As an example we shall code ‘hello world ’ with several techniques. With converse scanning . means downwards and – means to the right, so with Figure 1 –––– .– ....– ....– .–– – gives ‘hello ’ and ....–– .–– .––– ....– ...–– – gives ‘world ’, forty-four switches. With Morse code we find .... . .–.. .–.. ––– –– for ‘hello ’ and ––. ––– –.– .–.. –.. –– for ‘world ’, thirty-six switches. Codes for M and SP were exchanged. With two-bit quartering and Figure 11 :– .– :: :: .: + leads to ‘hello ’ and =–. .: –. :: :. + to ‘world ’, thirty-six switches as well. An animation is available, see Note.

Comparison by a simulated experiment

This section describes experiments to compare different scanning systems. Samuel Morse assigned shorter codes to more frequent letters and based himself on frequencies in printer's lay, shown in Table 1. Letter frequencies change little over time, and can be used to order letters in two-bit scanning. Optimal ordering depends on sums of frequencies as well as on user's expectations and is related to so-called Huffman code (Huffman, 1952).

A 8500	N 8000
B 1600	O 8000
C 3000	P 1700
D 4400	Q 500
E12000	R 6200
F 2500	S 8000
G 1700	T 9000
H 6400	U 3400
I 8000	V 1200
J 400	W 2000
K 800	X 400
L 4000	Y 2000
M 3000	Z 200

Table 1.Frequencies of characters in printer's lay, based on Stower, 1817.

We shall now use these frequencies to compare different input mechanisms in terms of the number of switches and pauses that users need to operate them. If each character c has a frequency f_c then Sum f_c is equal to the total number of all occurrences of the twenty-six characters in the alphabet. Sum f_c is 106900 in Table 1. If the number of switches to select character 'd' is called s_d, f_d s_d is equal to the number of switches used to select f_d occurrences of 'd'. In Figure 11 s_a=2 and according to Table 1 f_a=8500. Therefore f_a s_a = 8500 2 = 17000. Sum f_c s_c means the total number of switches to select all characters in Table 1 with a simulated experiment. The weighted average number of switches per character is (Sum f_c s_c)/(Sum f_c). These values can also be computed for other input mechanisms.

With two-bit quartering several codes are used to select a single character. Normally, each code is followed by a short pause. In Figure 11 the number of pauses per character is one for space, selected by +, two for frequent characters as 'a' or 'r' and three for less frequent characters as 'z'. With pause reduction the machine takes no pause after a code of length two such as =, :, + and ?. This means that a character such as 'v' can be selected from the scan matrix of Figure 11 with six switches and zero pauses, instead of six switches and three pauses. A character such as ‘a’ however will still require two switches and two pauses. The weighted average number of pauses per character was computed with frequencies of characters from Table 1 in the same manner as described above. See Table 2 for results.

	Switches	Pauses
Passive row-column scanning	3	3.29
Converse scanning one switch	3.29	2
Converse scanning two switches	3.29	1
Two-bit quartering	2.78	1.91
Two-bit quartering, pause reduction	2.78	1.04
Morse code	2.50	1

Table 2. Switches and pauses per character. Values computed with frequencies of letters A to Z as in Table 1 plus Space added with frequency of 26.700 or 20% of the total, ordering as in Figure 11, and Morse codes for M and SP exchanged.

Time per character

Table 2 allows us to estimate minimum times to key the average character with different input mechanisms that we may call k_{mc}, k_{cs} and k_{tbq}. If for some individual and some input mechanism s and p are the average switch time and pause time respectively, we may construct formulas like $k_{mc} = 2.50\ s + 1.0\ p$ and $k_{cs} = 3.29\ s + 1.0\ p$ and $k_{tbq} = 2.78\ s + 1.91\ p$. These are minimum times because users will also spend time to think, to verify, to decide and to look at the screen, activities that need not overlap with keying and that may require more time and attention. If we assume s=0.2 and p=0.5, certainly reasonable values, we get k_{mc}=1.0 seconds. Optimum speed with Morse code is now computed as 60/1.0=60 cpm (characters per minute). Likewise k_{cs}=1.66 seconds so optimum speed with converse scanning is 60/1.66=36.1 cpm and k_{tbq}=1.51 seconds so optimum speed with two-bit quartering is estimated as 60/1.51=39.7 cpm. Clearly, these values depend on the value of p, that may be influenced by training. With pause reduction the value for two-bit quartering is higher.

Optimal switch times and pause times probably differ between different input-mechanisms as well as between different people and have subtle interactions with user fatigue and with error rate. Still, these formulas, based on the simulated experiments of Table 2, make it plausible that after sufficient learning Morse code is faster than two-bit quartering with pause-reduction, which might be somewhat faster than converse scanning. It is not clear how much importance should be given to the rather long period of time that most people need to learn Morse code or to the fact that both Morse code and two-bit quartering can be easily executed with eyes closed.

Tests of learning

Five healthy paid test subjects learned two-bit quartering with an average of one hour practice. After one more hour of training they achieved speeds of up to twenty-six characters per minute in copy tasks maintained for up to thirty minutes and with eyes closed. No comparison was made with other input techniques. The author, who trained much more, achieves thirty-three characters per minute with pauses of 300 ms, a copy task and an error rate of less than 2%. These rates are higher than is to be expected with members of relevant patient groups, all extremely variable in terms of needs and of capacities.

A study was done with alternated use of three input techniques and a copy task of poetry. The single healthy subject received no instruction apart from a demonstration of 5 minutes. She was asked to alternate A, converse scanning with Figure 1, B, two-bit quartering with Figure 11 and C, Morse code with Figure 10 projected on screen. She exercised for about one hour then copied Dutch poetry for periods of 15 minutes with each input method and repeated the same poem once; AB, then another poem with CA, BC and so forth. Instead of switches she used two keys of a normal keyboard. Pauses were set at in all three conditions and changed after six sessions to respectively 620 ms in condition A and 320 ms, later 240 ms in condition B and C as optimal values for her. For results consult Table 3. Asked why she learned quickly she pointed out that she plays the piano. Her results suggest that some subjects should start with Morse code rather than either converse scanning or two-bit quartering.

Condition	Pauses 480 ms A 480 ms B&C		Pauses 320 ms B&C	Pauses 620 ms A 240 ms B&C	
A Converse scanning	22	9		19	23
B Two bit quartering	17	22	31	33	35
C Morse code	25	26	34	39	40

Table 3. Experiment with one healthy test subject aged 17 who copied poetry that was read to her. Each number represents characters per minute during one of fourteen sessions each of 15 minutes. The fourth session failed (9 cpm only), column represents session A and C with the same poem and the first session with C of the second poem.

An application with quartered scanning

Ultimately it is neither input rate nor effective communication that influences the quality of life for switch users but what they may do with it. Scanning mechanisms therefore have to be integrated in applications and in the opinion of the author strong reasons exist to combine them with stored text, wordprediction, macros and synthetic speech. This is a large subject on its own, see Lunn, Todman et al 2003 for an introduction and see Figure 12 for a screen image of such an aid with scanning as yet turned off.

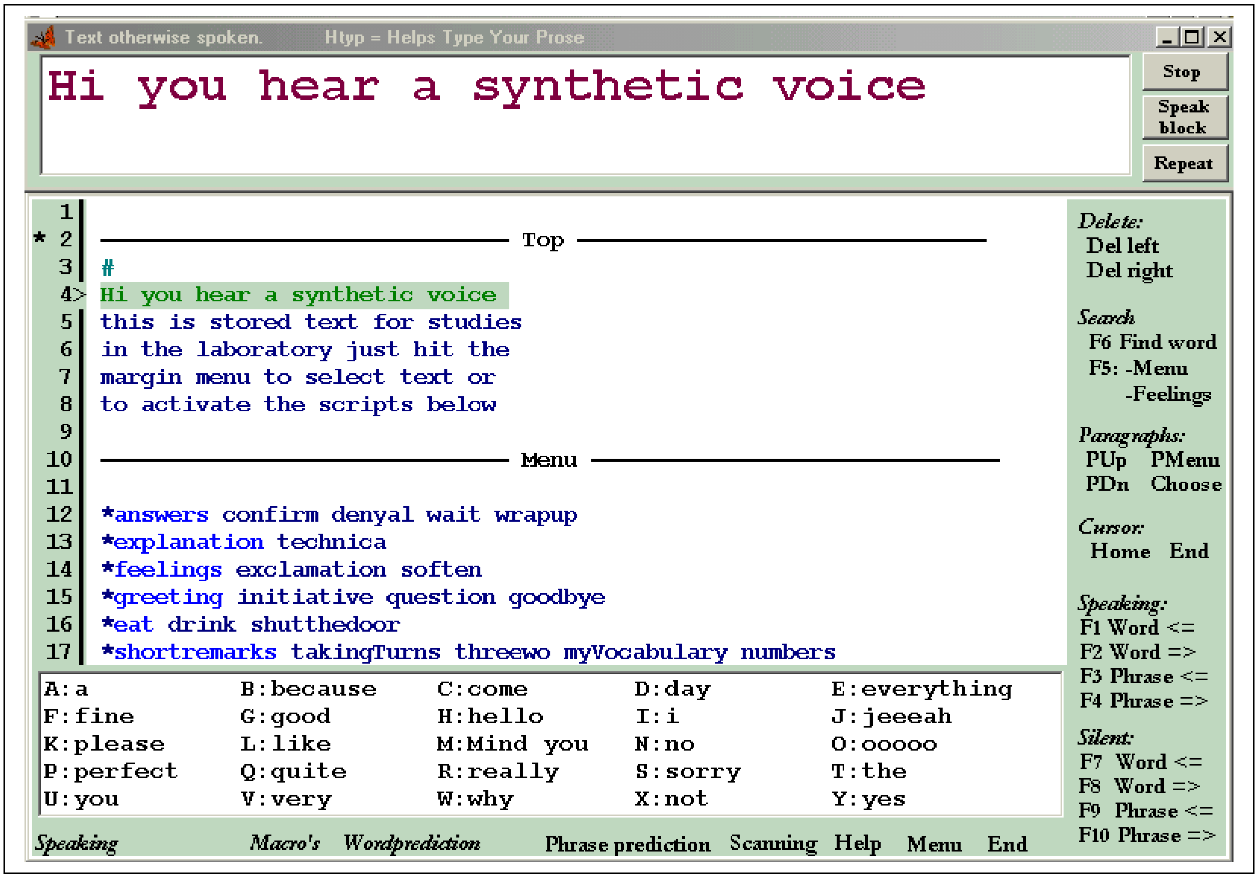


Figure 12. Text based communication aid with extremely rich interface.

The cursor finds itself at '#' and the mouse just did click on '4' to select the fourth line of text that was spoken. Lines twelve to seventeen start with an asterix and act as a simple scripts to paragraphs called '–Answers', '-Explanation' etcetera. If the mouse is pointed at '12' and clicked, paragraph '-Answers' is displayed. Function key F5 then moves to '-Confirm', the next paragraph in the script on line 12. In Figure 12 it moves to '–Menu' then '–Feelings' as is displayed in the right margin. Paragraphs can be useful if a conversation can be prepared, as to discuss what the doctor said or what to do over the weekend. Alas, for free flowing conversations stored text is an as yet unproven technique. 'b ' is redefined into 'because ' on line eighteen, one of twentysix lettermacros. Finally, on the last line, Functions like Macro's, word prediction, phrase prediction and scanning can be set. See Figure 13 for an image of a scan matrix after the first quarter is chosen.

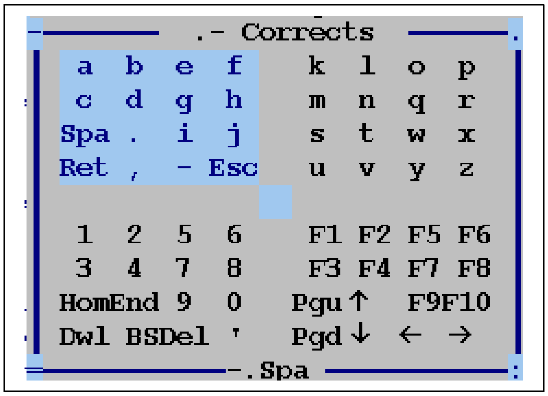


Figure 13. Scan matrix after '.' to select the first quarter.

If the optimised matrix of Figure 11 is used, access to word prediction is rather clumsy, as it requires long codes for 0..9 or shifting to another matrix to select a numeral. This can be alleviated by redefining codes of length 3 as shown in Figure 14.

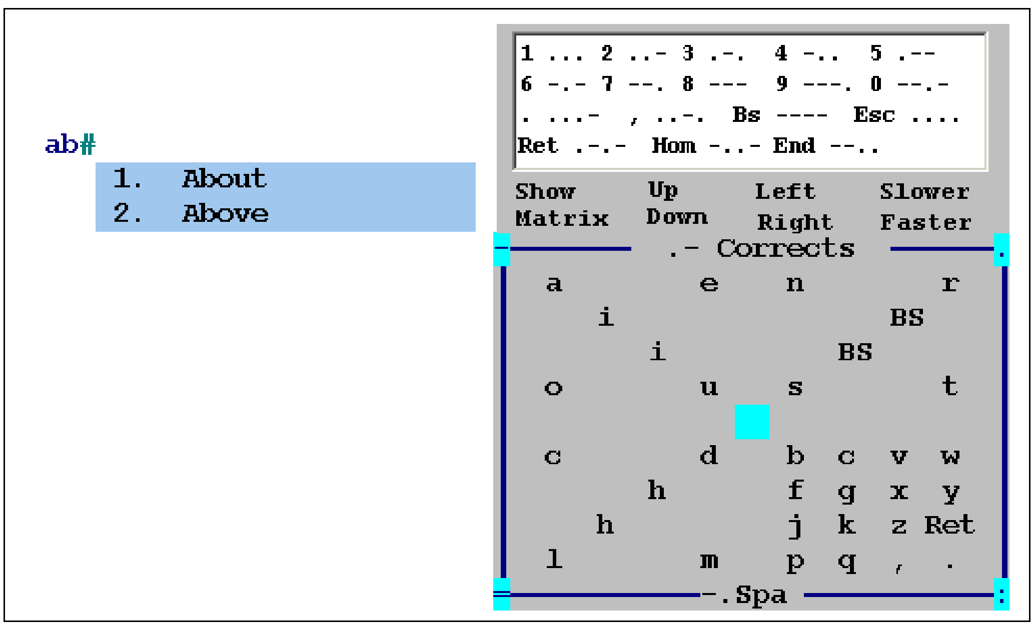


Figure 14.Scanmatrix with shortcuts and word prediction. The shortcut '-.' Or '+' is bound to Space. After '.' the first quarter is selected and the shortcut gets bound to 'i', presented twice in the matrix for technical reasons. ... acts as another type of shortcut, selecting '1' and therefore 'about' using word prediction.

To select the word 'about ' that is suggested by word prediction, a user may click '- - ' for 'a', ': - - ' for 'b' then ':. ' for '1', a total of 9 switches and 6 pauses. Remember that ':' means the same as '..' or two switches and no pause. Numerals 0 to 9 are redefined as short morse codes that act as bypassing codes in the scan matrix. Likewise, for mouse users, the right mouse button can be used to access word prediction from a distance, bypassing pointing actions. <mouse to 'a'><Left click><mouse to 'b'><Left click><Right click><Right click> selects the second word from the list with two mouse movements and four clicks only. In the laboratory and with extensive training impressive rates can be achieved with these techniques combined. Also, in roleplays, satisfactory communication was experienced about nine times out of ten: for more details contact the author. In how far they may improve perceived quality of life for patients with ALS requires clinical work and, quite possibly, linguistic and psychological tests as well.

Conclusion

Two-bit quartering is a new input method for switch users that is easy to learn and that may be applied within the field of Augmentative and Alternative Communication (AAC). For aphonic and dysarthric patients with Amyotrofic Lateral Sclerosis and the like it may be combined with text based communication aids that store text for later reuse. Patient studies are needed to determine if it adds to the quality of life of the intended user groups.

Note. A demo of two-bit quartering as well as of the text based aid of figure 12-14 is on http://depratendecomputer.nl where more background material can be found. Darci USB (Morse for Windows) is available from Westest Engineering (www.westest.com).

References

Damper, R.I. (1984). Text composition by the physically disabled: A rate prediction model for scanning input. *Applied Ergonomics*, 15, 4, 289-296.

Huffman, D. (1952). A method for the construction of minimum redundancy codes. *Proceedings IRE, 40,* pp. 1098-1101*, Sept. 1952.*

King, T. (2000). Modern Morse code in rehabilitation and education. New applications in assistive technology. Allyn and Bacon.

Lesher, G.W. & Moulton B.J. & Higginbotham D.J. (1998). Techniques for augmenting scanning communication. AAC, Augmentative and Alternative Communication. 14, June 1998, 81-101.

Lunn, J., Todman, J., File, P., Coles, E. (2003). Making Contact in the Workplace. Communication Matters National Symposium. Lancaster, 15-17 September.

Lynds, J. (2001). Darci for USB. Message on the morse2000 forum. See www.westest.com for more information.

Simpson, R. and Koester, H. (1999). "Adaptive One-Switch Row-Column Scanning," IEEE Transactions on Rehabilitation Engineering, 7(4), 464 - 473, December 1999.

Standage, T. (1998). The Victorian Internet. The remarkable story of the telegraph and the nineteenth century's online pioneers. Phoenix.

Stower C. (1817). ‘The Printer’s Manual: an abridgement of Stower’s Grammar.’ Boston:Crocker. Reprinted (1981) New York Garland. Quoted by Walsh (2000).

Vanderheiden, G.C. (1988). Overview of the basic selection techniques for augmentative communication. Past and future. In L.E. Bernstein (Ed.), The vocally impaired: Clinical practice and research (pp. 5-39). Philadelphia: Grune and Stratton.

Venkatagiri, H.S. (1999). Efficient keyboard layouts for Sequential Access. AAC, Augmentative and Alternative Communication 15, June 1999, 126-134.

Verrips, J. (2000). A communication aid with stored text. Int. J. of Rehabilitation Research, 23*,* 139-144*.*

Walsh, E.G. (2000). Mathematical analysis of Telegraphic Signalling. *MM71. September/October 2000.*