Dynamic Modelling of Keyboard Skills: Supporting Users with Motor Disabilities

Shari Trewin*, and Helen Pain

Department of Artificial Intelligence, University of Edinburgh, Scotland

Abstract. This paper describes the effective application of user modelling to the assessment of the physical ease with which a user can operate a standard QWERTY keyboard. The application is unusual in the sense that physical rather than cognitive skills are being modelled. The model examines four important skills which a user may have difficulty with, and produces an assessment of the ideal keyboard configuration for that user. This assessment can then be used to adapt the keyboard. For users with motor disabilities, such adaption can minimise or even eliminate the problems they experience. The model dynamically adapts to the current user and operates on free English text input. It has been evaluated using typing data from twenty keyboard users with disabilities and six without. The configuration recommendations made are very well matched to the users' problem areas.

1 Introduction

Computer users with motor disabilities can experience difficulties with the operation of QWERTY keyboards. If we were able to identify and model the specific difficulties of individual users, we could then use such models as the basis for recommendation of a more appropriate keyboard configuration for each user. We believe that this would make the keyboard easier to use, and reduce the number of errors occurring. This paper describes the development and evaluation of techniques for identifying and modelling keyboard difficulties. Our focus is on the modelling of physical skills, rather than on cognitive skills.

Although alternative input devices (such as switches) are available, many users with disabilities find that keyboards provide a more efficient input device. Errors that occur through physical difficulty in manipulating the keyboard are referred to here as *performance errors*. Empirical research with keyboard users with disabilities has highlighted six common classes of performance error (Trewin and Pain, 1996a). These are:

1. *Long Key Press Errors*: An alphanumeric key is unintentionally pressed for longer than the default key repeat delay. On the majority of operating systems, there is a *Repeat Keys* facility, which allows the user to control the length of time a key must be held down for before it repeats. Setting an appropriate delay, or disabling key repeats altogether, can prevent long key press errors.
2. *Dropping Errors*: The user fails to press two keys simultaneously (e.g. use of the *Shift* key). This error type is just one manifestation of difficulty in pressing down two keys at once. The *Sticky Keys* facility, when activated, causes modifier keys to latch. When pressed, they stay active until the next key has been pressed. With *Sticky Keys*, a user presses *Shift* and then 'a' to produce a capital 'A'. Use of *Sticky Keys* can eliminate dropping errors.

* The authors acknowledge the support of the University of Edinburgh in funding this research.

3. *Additional Key Errors*: A key adjacent to the intended key is activated. In the majority of such errors, both the intended and additional keys are pressed, and the key presses overlap in time. For this class of additional key errors, the *Overlap Keys* configuration facility has been proposed (Trewin and Pain, 1996b). This would inhibit one or both of the overlapping keystrokes, and may be useful for keyboard users who do not overlap keystrokes in their normal typing.
4. *Bounce Errors*: The user unintentionally presses the intended key more than once. These errors are targeted by the *Bounce Keys* facility, available on many operating systems. *Bounce Keys* introduces a delay after a key press, during which time the same key cannot be reactivated. The length of the delay can be adjusted.
5. *Missing Key Errors*: The user fails to activate their intended key, either because they missed it, or because they did not press it hard enough. No existing keyboard configuration facility can alleviate missing key errors.
6. *Remote Key Errors*: The user unintentionally activates a key remote from any key they intended to activate. This can happen if, for example, a user leans on some part of the keyboard. No existing software configuration facility can alleviate remote key errors.

While it may be possible to reduce or eliminate these errors by using specialised keyboards, current device selection methods are subjective and time consuming, involving much trial and error (Broadbent and Curran, 1992, Casali, 1995, and Smutz et al., 1994). Standard keyboard configuration facilities can be tried immediately, cost nothing, and do not restrict the user to a single machine. Often they are sufficient to allow good keyboard access.

Unfortunately, users are frequently unaware of facilities that could be used to customise their particular environment, for example setting appropriate values for the key repeat delay. Rather than flooding the user with information about what they might do, a user model can be used to focus on those facilities which are most relevant to their specific needs. Such a model can be developed from monitoring the user's typing skills, and identifying their performance errors.

As Self (1988) has pointed out, there is little point in modelling problems for which there is no solution. Consequently, our model focuses on the first four error types listed: those for which existing or proposed keyboard configuration facilities provide support.

The following section discusses established user modelling techniques, and explores why, despite its good fit with the high level goals of a user model, this domain is unsuitable for the majority of existing techniques.

2 User Modelling Techniques

The problem of choosing an appropriate configuration is in many respects a traditional user modelling problem. It fits the definition of "the knowledge and inference mechanism which differentiates the interaction across individuals", suggested by Allen (1990). The system is required to adapt to individual users whose requirements vary enormously, and it takes responsibility for ensuring successful system-user communication (see Rich, 1983, and Kass and Finin, 1988). By Rich's classification, it is individual and implicit: that is, it performs its own customisation through choosing an appropriate set of configuration options to suggest. The target users are not able to choose their own set of configuration options, hence an explicit model is of little use to them.

The model used by a configuration support application must be dynamic (as defined by Kass and Finin, 1989): it must be able to adjust to the changing requirements of users, which may vary greatly according to factors such as fatigue. It must also adapt to different users who may be using the same computer, where there may be no explicit indication of the change of user. Further requirements are that the model should be unobtrusive and general, so that users are not required to perform specific tasks in order for their keyboard skills to be assessed. Ideally assessment should take place during their normal typing, potentially allowing a large volume of typing data to be examined.

Despite the problem of interest being characterised as a traditional user modelling problem, many of the common techniques used are not suitable here. Those that rely on stereotyping (Rich, 1989) are not applicable: similar keyboard problems may stem from very different disabilities, and similar disabilities may produce very different performance errors.

Approaches using bug libraries and overlay models, such as those used in intelligent tutoring systems (see Clancey, 1987, and Brown and Burton, 1978) are also either too restrictive or inappropriate. These models capture information about how a student's skills and knowledge differ from those of an expert, and are dependent on knowing the user's task, and either identifying missing knowledge or hypothesising about the reason for any incorrect answers. Since free text is permitted, a mechanism reliant on knowing the text a user was trying to type would be too restrictive. For similar reasons, feature based modelling (Webb and Kuzmycz, 1996) is also inappropriate. A further constraint on such approaches is their assumption of consistency in the user's behaviour. Keyboard errors are highly inconsistent, in that they do not occur at every possible opportunity: not every key press will be too long, for example. It is the frequency of errors that indicates those with genuine difficulties. Even in teaching domains, this assumption can cause problems (Self, 1988). Any technique for modelling keyboard skills must deal in frequencies, rather than binary values like known/not known.

The model should also be capable of managing uncertainty over the classification of a character sequence as being correct or containing some performance error. Uncertainty arises here because the user's task is unknown. Established numerical techniques for managing uncertainty – Bayesian networks and Dempster-Schafer theory (Jameson, 1996) – are not ideal. Bayesian networks could in principle be applied, but the full power of this technique is not required, due to the small number of sources of evidence available. Similarly, the ability of Dempster-Schafer theory to combine pieces of uncertain evidence is also not required, as the information sources available are reliable. Given the simplicity of the data available, less complex (and less theoretically motivated) criteria have proved adequate for decision-making.

Because of the uncertainty in the interpretation of an input stream, the model must be tolerant of errors in the performance error recognition mechanisms. It must also be sensitive to medium term variations in the user's typing characteristics, so that the configuration can be altered as the user's requirements change. The following section outlines the approach taken.

3 Recognising Keyboard Difficulties

The model of typing abilities focuses on the four classes of performance error for which some compensatory mechanism exists or has been proposed. Investigation of these areas is carried out unobtrusively by trapping and examining keyboard events before they are passed on to the application in use.

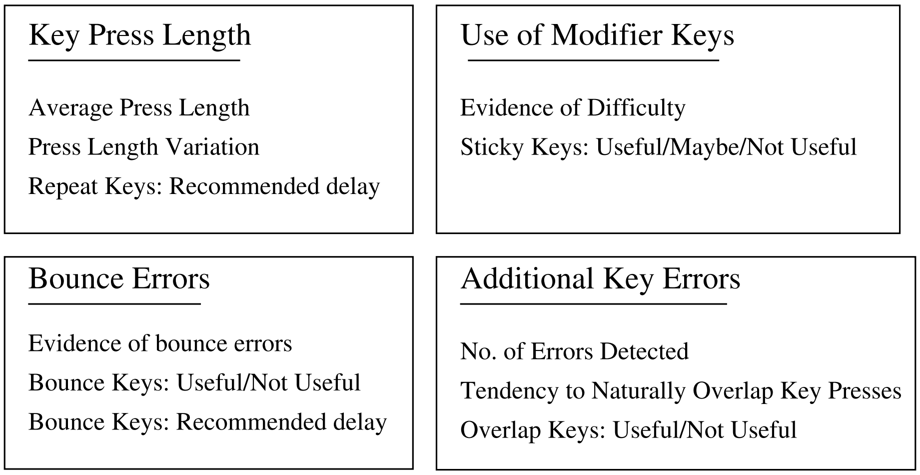

Figure 1. Outline of the user model.

The user model itself is outlined in Figure 1. It contains both general information about the user's typing characteristics and specific information about the recommended keyboard configuration for the current user. The model is dynamically updated as evidence about the current user's typing abilities is gathered. Threshold values and decay of evidence over time are used to damp out the effect of small variations in typing style, and of any errors made in the recognition of difficulties. Note that no changes are made to the actual keyboard configuration in use – the model simply makes recommendations.

Throughout the assessment of keyboard difficulties, an assumption is made that the user is typing English text, probably in a word processing application. The model uses a database storing the frequency with which a given character is followed by another given character in modern English.[1] The frequencies were calculated from the British National Corpus, which contains over 100 million words, representing many different varieties of English. (More information is available at: `http:info.ox.ac.uk/bnc`.) The digram information could be replaced or supplemented with similar statistics about languages other than English, or any command or programming language.

The key repeat delay chosen for the current user is based on their average key press length, and the amount by which their key presses tend to vary upwards from that average value. These calculations are limited to those keys which are rarely deliberately repeated. For example the *Backspace* key is often held down for a long period in order to delete a sequence of characters, and is excluded from the calculation. In addition, abnormally long key presses are ignored, on

[1] The use of digram frequencies, as opposed to a dictionary, has a number of advantages. It eliminates effects due to misspellings in other parts of a word, or words not in the dictionary, and can also handle errors involving the space bar. Digram lookup is also faster than dictionary search, and requires less memory. Speed of classification is important, since the model should not visibly affect the response time of the user's application.

the basis that they are likely to be deliberate, or caused by an event such as the user leaning on the keyboard. The recogniser chooses a value which is longer than approximately 98% of key presses. If large numbers of abnormally long key presses are observed, it is recommended that the repeat facility should be disabled.

Assessment of the user's ease of use of modifier keys is based on the observation that, in the data available, subjects who had difficulty in pressing two keys at once would often type characteristic keystroke sequences, or adopt specific strategies for avoiding multiple key presses. Recognition of difficulties in pressing multiple keys at once is based on the detection of such patterns, and these are weighted according to the strength of the evidence they provide. Indicative patterns include:

- Use of *Caps Lock* for a single key press.
- Pressing a modifier key, followed by a small letter, followed by the *Backspace* key.
- Starting a sentence with a small letter.

In the vast majority of the 163 additional key presses observed in the data available, the unintended key press overlapped in time with that of the intended key. Given this observation, all overlapping keystrokes are candidate additional key errors. Using knowledge of the keyboard layout, English digram frequencies, and the current user's typing style, each overlap is classified as deliberate, an error, or of unknown cause. In the data available, 77% of the subjects rarely or never deliberately overlapped keystrokes, so the user's typing style is an important source of information in this process. The current keyboard layout is that of a Macintosh QWERTY keyboard, but other keyboards could easily be used.

Detection of bounce errors is the most difficult of the four areas tackled by the model. Many people who make bounce errors are also capable of fast deliberate double key presses. The recogniser therefore has two challenges: to spot people who are making bounce errors, and to select a delay which will minimise deliberate key press loss, while eliminating as many bounce errors as possible.

The recogniser operates by examining all double letters and assessing their likelihood of being bounce errors. Knowledge of word processing, English and the timing of the double letter is used. For each double, an evidence value between zero and ten is calculated. The greater the value, the higher the system's confidence that a bounce error has occurred. The choice of value for the delay to be imposed is conservative, preferring to miss some bounce errors rather than eliminate deliberate double key presses.

4 Evaluation

Evaluation of the model is based on typing data gathered from an empirical study of the keyboard difficulties experienced by people with motor disabilities, described by Trewin and Pain (1996a). Twenty subjects with motor disabilities and six without were asked to type a set passage twice. The passage was approximately 100 words (547 characters) long and required 25 uses of a modifier key. The errors made were established through direct observation and video evidence, while a detailed log recorded the `KeyDown` and `KeyUp` events reported to the computer, including timings measured in *ticks* (sixtieths of a second). Macintosh computers and the ClarisWorks word processing package were used.

From this study, 44 log files were available. These were used to simulate direct computer input – reading from the file instead of the event queue.[2] When the whole log file had been read, the state of the user model was examined. For long key press errors, additional key errors and bounce errors, the accuracy of the model's configuration recommendations is assessed by examining the number of errors occurring in each typing test, and where possible comparing this with an estimate of the number of errors that would have occurred had the recommended configuration been used. For modifier key usage, a more sophisticated approach is required, to take into account the coping strategies adopted by users who find it difficult to press two keys at once. Assessment of the model in this area is based not only on error numbers, but also on the user's reported and observed ease of using both hands, and their preference (if known) for using *Sticky Keys*.

The model is text-independent, but because the evaluation data consists of many copies of the same text passage, further evaluation of the modelling techniques in real situations is necessary to increase confidence in the accuracy of the model over general English text.

The following sections describe the results achieved by these techniques in each of the four areas of keyboard difficulty where configuration may be helpful.

5 Detection of Long Key Press Errors

Long key press errors were the most common type of difficulty found in the original study, and choosing an appropriate setting for the key repeat delay is for many the single most important mechanism for improving keyboard usability.

Table 1 shows the results of the model for the twenty subjects with disabilities (1-20) and the comparison group (C1-C6). The table shows each subject's reported level of difficulty in making quick key presses, the repeat delay setting (measured in ticks) recommended by the recogniser for each of their two typing tasks (T1 and T2), the number of long key press errors that would have occurred in each task had the recommended repeat delay been in force for the whole of the time spent typing, and the average key press length.

The recommended delays ranged from 10 to 41 ticks, and the maximum number of long key press errors remaining in a single task was 17, for Subject 17, which represents a 2.8% error rate. In the original data, the maximum error rate for a default repeat delay of 16 ticks was 66.6%, for Subject 13.

Six of the subjects reported some difficulty in making short key presses, and indeed the three subjects for whom the longest repeat delays were suggested were among this group. A total of 17 subjects, including one from the comparison group (a novice computer user), were advised to use key repeat delays longer than the default.

One interesting result is that for Subject 5, who rated short key presses as 'easy'. In both her typing tasks the average key press length was 5 ticks, but the variation among key press lengths was large. She made many long key presses, particularly in the second task (fatigue may have contributed to the difference). Because the recogniser takes into account this variation, long repeat delays are advised in order to cope with the longer key presses she sometimes makes. A

[2] In a real situation, dynamic events rather than a log file would be used. This avoids potential misuse of logs for the assessment of typing productivity.

Table 1. Long key press recognition.

Subject	Reported Difficulty	Setting Chosen T1	Setting Chosen T2	Errors Remaining T1	Errors Remaining T2	Average Key Press Length
1	easy	15	18	1	0	7
2	easy	12	-	0	-	5
3	easy	21	21	4	5	9
4	easy	11	11	1	2	4
5	easy	19	30	11	13	6
6	easy	24	22	12	10	10
7	some difficulty	24	25	11	11	11
8	easy	22	-	15	-	12
9	easy	19	21	9	12	9
10	hard	38	-	5	-	17
11	easy	11	11	3	2	4
12	easy	35	32	1	0	16
13	very hard	41	-	0	-	20
14	hard	22	-	1	-	10
15	hard	21	23	2	0	10
16	easy	12	12	0	0	5
17	easy	26	-	17	-	10
18	easy	20	-	0	-	9
19	extreme	34	36	1	0	16
20	easy	24	23	1	1	10
C1	easy	19	-	0	-	8
C2	easy	11	12	0	0	5
C3	easy	11	11	0	0	4
C4	easy	12	12	0	0	5
C5	easy	10	11	0	0	4
C6	easy	13	13	0	0	5

recogniser based purely on average key press lengths would be unable to accommodate subjects with similar wide variations in their key press lengths.

The projected total number of long key press errors for all subjects using their recommended setting is 151, as opposed to the 2610 projected errors under a default key repeat delay. This represents a significantly improved individual configuration for the subjects studied.

6 Detection of Problems Pressing Two Keys at Once

The results of modelling difficulties in the use of modifier keys are summarised in Table 2. The table shows, for each subject,[3] the difficulty they reported in performing multiple key presses, the

[3] The comparison group are included in this analysis but excluded from the table – none had difficulty in using modifier keys, and *Sticky Keys* was not suggested or recommended for any of them. Only one dropping error occurred, and very little evidence of difficulty was gathered.

number of dropping errors they made in each typing task (T1 and T2), the final total of accumulated evidence of a need for *Sticky Keys*, the *Sticky Keys* setting recommended by the recogniser ('on', 'off' or 'maybe'), and the setting that would have been chosen for each subject in a real situation. This last value was arrived at by considering whether the subject is a predominantly one-handed typist, their reported level of difficulty, their preferred configuration, and the number of dropping errors they made.

Table 2. Modifier key difficulty recognition.

Subject	Reported Difficulty	Dropping Errors		Sticky Keys Evidence		Recommended Setting		Ideal Setting
		T1	T2	T1	T2	T1	T2	
1	impossible	0	3	47	81	on	on	on
2	easy	0	-	7	-	off	-	on
3	easy	0	1	0	0	off	off	off
4	moderate	8	6	25	19	maybe	maybe	maybe
5	hard	0	0	1	1	off	off	on
6	impossible	2	3	70	60	on	on	on
7	easy	4	0	22	4	maybe	off	maybe
8	easy	0	-	2	-	off	-	off
9	some difficulty	0	2	3	6	off	off	maybe
10	very hard	0	-	71	-	on	-	on
11	impossible	0	0	84	48	on	on	on
12	some difficulty	4	2	54	53	on	on	on
13	easy	3	-	14	-	maybe	-	maybe
14	easy	0	-	0	-	off	-	off
15	hard	11	2	36	61	on	on	on
16	easy	0	0	9	0	off	off	off
17	easy	0	-	0	-	off	-	off
18	easy	0	-	43	-	on	-	off
19	moderate	0	0	11	0	maybe	off	maybe
20	moderate	2	2	58	14	on	maybe	on

The configuration recommended agrees with the 'ideal' configuration for 22 of the 26 subjects. Of the four cases where the recogniser made a less than ideal choice, three were subjects for whom *Sticky Keys* might be useful, but for whom it was not recommended. All of these subjects were able to use modifier keys competently, but typed predominantly with one hand. The recogniser cannot detect how awkward or tiring an action may be for the user, and can only judge their actual performance.

In the remaining case, that of Subject 18, the model mistakenly recommended the use of *Sticky Keys*. This recommendation was based on the observation that Subject 18 used the *Caps Lock* key for all single capital letters. This was actually due to a lack of understanding of the keyboard, rather than difficulty with modifier key presses. Cases such as this may be common, and the possibility that the user does not understand the use of *Caps Lock* or *Shift* should be considered by any system interpreting these recommendations.

Overall, the performance of the model is good. The use of *Sticky Keys* was recommended for all those who rated modifier key presses as very hard or impossible.

7 Detection of Additional Key Errors

Additional key errors were relatively common in the data available, and usually involved two key presses which overlapped in time. Table 3 shows, for each subject, their reported ease of isolating keys to press (Reported Difficulty), whether a problem with additional key errors was found by the model (Problem Indication), and the actual numbers of additional key errors made in the tests (Additional Errors) .

Table 3. Additional key error recognition.

Subject	Reported Difficulty	Problem Indication		Additional Errors		Number Detected		Spurious Errors		Deliberate Overlaps
		T1	T2	T1	T2	T1	T2	T1	T2	
1	moderate	1	1	29	29	23	24	0	2	no
2	easy	0	0	0	-	0	-	0	-	no
3	some difficulty	0	0	5	4	2	2	0	0	yes
4	some difficulty	0	0	3	0	1	0	0	1	no
5	some difficulty	0	0	2	0	2	0	0	0	no
6	easy	0	0	0	1	0	1	0	0	no
7	some difficulty	0	0	3	0	0	0	0	0	no
8	easy	0	0	3	-	1	-	0	-	no
9	easy	0	0	2	0	2	0	0	1	a little
10	easy	0	0	3	-	2	-	0	-	no
11	easy	0	0	3	1	3	0	0	0	no
12	easy	0	0	2	0	0	0	0	0	no
13	easy	0	0	0	-	0	-	0	-	no
14	some difficulty	0	0	0	-	0	-	0	-	no
15	some difficulty	0	0	5	7	3	3	1	0	no
16	easy	0	0	1	0	0	0	0	0	no
17	easy	0	0	4	-	4	-	0	-	no
18	easy	0	0	0	-	0	-	0	-	no
19	moderate	1	0	18	11	8	3	1	2	yes
20	easy	1	1	14	9	10	8	0	0	no
C1	easy	0	-	0	-	0	-	0	-	no
C2	easy	0	0	1	0	0	0	3	2	yes
C3	easy	0	0	0	2	0	0	0	0	a little
C4	easy	0	0	1	0	1	0	1	1	a little
C5	easy	0	0	0	0	0	0	1	0	no
C6	easy	0	0	0	0	0	0	0	0	no

A problem was identified for three subjects. These were the three subjects who made the most additional key errors, including the two who reported the most difficulty. The majority of the remaining subjects did make some additional key errors, but their error rates were low.

The table also shows the number of genuine errors that were detected (Number Detected) – 63% of those present in the original data. Errors are missed most frequently for those subjects who often deliberately overlap keystrokes. These are indicated in the final column of the table (Deliberate Overlaps). For these subjects, particularly Subject 19, additional key errors are difficult to distinguish from normal typing. The error detection mechanism is conservative, in order to avoid detection of errors where none exist. Nevertheless, 16 spurious errors (Spurious Errors) were found, also shown in the table.

To allow for spurious errors, a rate of one or two errors in every 100 characters is tolerated. Error rates above this threshold will cause a problem to be flagged. For some users who make additional key errors, *OverlapKeys* may provide a useful level of support.

Table 4. Bounce error recognition.

Subject	Bounce Key Setting		Bounce Errors		Bounce Evidence	
	T1	T2	T1	T2	T1	T2
1	off	off	0	0	0.0	0.0
2	off	-	0	-	0.0	-
3	off	off	0	0	0.0	0.7
4	off	off	0	0	0.0	0.0
5	off	off	0	0	0.0	0.0
6	2	2	8	12	17.9	6.9
7	off	off	0	0	0.0	2.3
8	off	-	0	-	0.0	-
9	off	3	0	1	0.8	5.8
10	off	-	0	-	0.0	-
11	3	off	0	0	7.1	0.0
12	off	off	2	1	0.0	1.0
13	6	-	3	-	7.9	-
14	off	-	0	-	0.0	-
15	5	off	3	3	16.1	0.0
16	off	off	0	0	1.6	0.0
17	off	-	0	-	2.8	-
18	off	-	0	-	0.0	-
19	off	off	3	0	1.8	0.0
20	5	off	5	3	18.5	3.2
C1	off	-	0	-	0.9	-
C2	off	off	0	0	0.0	2.3
C3	off	off	0	0	0.0	0.0
C4	off	off	0	0	1.6	0.1
C5	off	off	0	0	1.8	0.0
C6	off	off	0	0	0.0	0.0

8 Detection of Bounce Errors

Seven subjects made up the total of 44 bounce errors. The results of bounce error detection are shown in Table 4. The use of *Bounce Keys* is recommended in six of the eleven tasks in which at least one bounce error occurred. Six bounce errors were on the *Caps Lock* key, including all three of the errors in Subject 15's second typing task. Bounce errors on *Caps Lock* cannot be detected by this mechanism, or eliminated by the use of *Bounce Keys*.

Bounce Keys is also recommended for one subject who did not make bounce errors. In this case, a high level of spurious evidence had been gathered. A longer typing test is necessary to reveal whether this evidence would decay into insignificance, or whether similar results would develop for other subjects.

The *Bounce Keys* settings chosen by the model varied between 2 and 5 ticks. Imposing these recommended delays on reactivation of keys would have eliminated 15 of the 38 bounce errors not on the *Caps Lock* key. They would also eliminate 5 deliberate key presses. It is difficult to separate deliberate key presses from bounce errors, and so the results here seem a good compromise – losing one deliberate key press for every three errors eliminated.

9 Summary

We have developed a model of a user's keyboard abilities in four important areas. In all of these areas, existing or proposed keyboard access facilities can alleviate or eliminate difficulties that a user with motor disabilities may experience. The model uses simple statistical techniques. Unlike many traditional user modelling techniques, it has no knowledge of what the user is attempting to type. The solutions are dynamic, user-specific and unobtrusive.

The accuracy of the model has been evaluated using a set of 44 recorded typing logs, made by twenty users with motor disabilities, and six without. Evaluation on dynamic typing data is in progress. If the recommendations made by the model were applied to the original logs, the chosen *Repeat Keys* settings could have reduced the number of long key press errors from 2610 to 151. Use of *Sticky Keys* where recommended could have eliminated 54 of the 56 dropping errors, and would have helped 9 of the 11 subjects who reported difficulty in using modifier keys. The use of *Overlap Keys* was suggested for all 3 subjects prone to additional key errors, and the use of *Bounce Keys* was suggested for 5 of the 7 subjects who made bounce errors. Throughout the four areas and 44 tasks, in only two cases did the model recommend the use of an unnecessary facility. One of these was due to the subject's misunderstanding of the use of modifier keys.

The model we have developed makes explicit configuration recommendations, on which a user is free to act. It does not, however, offer the user any support with understanding, finding and setting the recommended options. While simple recommendations leave the user in control, some users may be unable to alter their configuration themselves. The authors' current work is investigating the feasibility of an adaptive configuration support system incorporating this model, and the use of such a system to actively help keyboard users with motor disabilities to find and set up the keyboard configuration that best suits their needs.

References

Allen, R. (1990). User models: Theory, method and practice. *International Journal of Man-Machine Studies* 32:511–543.

Broadbent, S., and Curran, S. (1992). *The Assessment, Disability and Technology Handbook*. Oldham: North West Regional ACCESS Centre and Oldham Education Department.

Brown, J., and Burton, R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science* 2:155–192.

Casali, S. (1995). A physical skills based strategy for choosing an appropriate interface method. In Edwards, A. D. N., ed., *Extra-Ordinary Human-Computer Interaction: Interfaces for Users with Disabilities*. Cambridge University Press. chapter 17, 315–341.

Clancey, W. (1987). *Knowledge-Based Tutoring: The GUIDON Program*. MIT Press.

Jameson, A. (1996). Numerical uncertainty management in user and student modeling: An overview of systems and issues. *User Modeling and User-Adapted Interaction* 5:193–251.

Kass, R., and Finin, T. (1988). A general user modelling facility. In *Proceedings of Computer Human Interaction*, 145–150. New York: ACM.

Kass, R., and Finin, T. (1989). The role of user models in cooperative interactive systems. *International Journal of Intelligent Systems* 4:81–112.

Rich, E. (1983). Users are individuals: Individualising user models. *International Journal of Man-Machine Studies* 18:199–214.

Rich, E. (1989). Stereotypes and user modeling. In Kobsa, A., and Wahlster, W., eds., *User Models in Dialog Systems*. Berlin: Springer-Verlag. 35–51.

Self, J. (1988). Bypassing the intractable problem of student modelling. In Gauthier, G., and Frasson, C., eds., *Proceedings of Intelligent Tutoring Systems '88*, 18–24.

Smutz, P., Serina, E., and Rempel, D. (1994). A system for evaluating the effect of keyboard design on force, posture, comfort and productivity. *Ergonomics* 37(10):1649–1660.

Trewin, S., and Pain, H. (1996a). Keyboard and mouse errors due to motor disabilities. DAI research paper 838, AI Dept, University of Edinburgh. Submitted for publication.

Trewin, S., and Pain, H. (1996b). On the adequacy and uptake of keyboard access facilities for people with motor disabilities. DAI research paper 839, AI Dept, University of Edinburgh. Submitted for publication.

Webb, G., and Kuzmycz, M. (1996). Feature based modelling: A methodology for producing coherent, consistent, dynamically changing models of agents' competencies. *User Modeling and User-Adapted Interaction* 5:117–150.