Number Entry Interfaces and their Effects on Error Detection

Patrick Oladimeji[1], Harold Thimbleby[1] and Anna Cox[2]

[1] Future Interaction Technology Lab, Swansea University, SA2 8PP, UK
[2] UCL Interaction Center, University College London, WC1E 6BT, UK
{p.oladimeji, h.thimbleby}@swansea.ac.uk, anna.cox@ucl.ac.uk

Abstract. A significant amount of interaction involves number entry. The purpose of any number entry interface is to accurately select or set a numeric value. There are two main styles of number entry interfaces found on medical devices: serial interfaces like the ubiquitous 12-key numeric keypad, and incremental interfaces that use a knob or a pair of keys to increase or decrease numbers. We report an experiment that investigates the effect of interface design on error detection in number entry. The initial findings show that the incremental interface produces more accurate inputs than the serial interface, and the magnitude of errors suggests that the incremental interface could reduce the death rate relative to the numeric keypad.
Keywords: number entry, data entry, error detection

1 Introduction

Number entry is perceived to be a very simple and mundane task — yet numerical drug dosing errors account for a significant portion of adverse drug events (ADEs) in hospitals, particularly in paediatrics [2, 5]. Number entry errors can be as a result of a combination of user errors, poor interaction design and hardware defects. Hardware defects on keypads (e.g., key bounces) have been reported by the Institute for Safe Medication Practices (ISMP) as a source of error in medical device programming [1]. A key bounce occurs when physically pressing a key once causes a repeat of the same key; this is different from a double keying error where a user accidentally presses the same key twice. If a key bounce is undetected, it might lead to an overdose, which in turn might lead to patient harm or death.

In many user interfaces, number entry is implicit; for example, adjusting sound levels by rotating an unmarked dial, or moving a scroll bar adjusts a hidden number — but the user copes because of direct feedback (direct manipulation). In this paper we are concerned with numbers that are displayed by the user interface as precise Arabic numerals (e.g., 123, 6.5, etc) rather than as abstract values (e.g., colors), or even as numbers indicated by pointers on numeric scales (e.g., analog meters). This style is prevalent in safety-critical applications such as drug rate control in healthcare.

Number entry interfaces may be grouped into two main categories:

- **Serial number entry:** A familiar example of serial entry is the 12-key numeric keypad used on calculators. The user enters the desired number typically using a

keypad. The number is entered serially, digit by digit starting from the leftmost digit. There are two main variations: those with a decimal point key and those without. When there is no decimal point key, a decimal point is inserted in a fixed position, typically giving 2 decimal digits (so keying 1234 would input 12.34). Serial entry provides the quickest method for arbitrary number entry, and its use also coincides with how numbers are spoken in many Western languages.

- **Incremental number entry:** In an incremental interface, the user is only able to increment or decrement the number, using fixed actions, typically a pair of up/down keys, rotary dials, or pressure sensitive controls. As a result, the number entry is about changes to a current number. The user may be able to influence the rate of change of the number by exerting more pressure on the control (in the case of a pressure sensitive interface) or by the speed with which the control is manipulated (e.g., turning a knob), or by the length of time the interaction is prolonged for (e.g., holding down an increment key longer to invoke faster or larger changes on a number). User interfaces typically refine these styles, for instance to impose numerical limits and by varying key layout.

Unfortunately, errors are eventually inevitable when using interactive systems and these can be in the form of mistakes, slips or lapses [7]. Sometimes, errors are detected by users and corrected. When errors go undetected, the consequences can be very serious. In a safety critical and dependable system, it is important that users realise when they commit errors and correct the errors. We believe that the differences in the design of number entry user interfaces place different demands on users in terms of what part of the user interface they focus most of their attention on and as a result whether they notice that an error has occurred and correct the error.

In this paper, we investigate the behaviour of users performing a routine task of number entry using serial and incremental user interfaces. We report our findings on the relative accuracies of both styles and a classification of the types of errors committed by users.

2 Experiment

Because interaction on the incremental interface requires users to monitor how the value on the display changes based on what key the user is interacting with, we hypothesize that users will more likely detect and correct errors when using an incremental interface as opposed to when using a serial interface. We also anticipate that users will pay more visual attention to the display than to the input when using the incremental interface and pay more visual attention to the input than to the display when using the serial interface. Finally, we expect that there will be types of errors that are unique to each class of interface.

Fig. 1. Screen shot of the incremental interface tested in the experiment.

2.1 Participants

22 participants (12 female) aged 18–55 years took part in the experiment. All participants were regular users of computers. None were prone to repetitive strain injury. One was dyslexic. Each participant was compensated for their time with a gift voucher.

2.2 Apparatus

A computer with an integrated Tobii eye-tracker was used to present the instructions and the number entry interfaces. Interaction with the computer was with a mouse, which was used to click on "keys" on the screen. 100 numbers were generated randomly for the experiment with the following constraints: all numbers were between 0 and 10, all numbers had a decimal point, all numbers had at least one non-zero significant digit after the decimal point and all numbers were different.

The serial interface was based on the Graseby 500 infusion pump. It allowed number entry using a full numeric keypad in the calculator style layout. This interface had a decimal point key and had a cancel key for deleting the rightmost character on the display. This interface allowed a maximum of 5 characters in its display, which may include only one decimal point.

The incremental interface was based on the Alaris GP infusion pump. It had four keys (Fig. 1.). The first two keys, with the upward pointing chevrons, increased the value displayed and the last two keys decreased the value. For each of the two sets of keys, one key (with the double chevron) caused a bigger change while the other caused a change that is ten times smaller. This interface allowed two modes of interaction. The user could click the keys or press and hold the keys. Clicking the keys changed the displayed value as specified above. Pressing and holding the keys changed the displayed value at a rate proportional to the duration the key was held down for. Users were expected to press and hold for faster increments or decrements. This interface always showed a decimal point and two digits after the decimal point.

A key bounce was triggered for the 84th, 88th, 92nd and 97th entry for both interfaces to see if users will detect and correct the errors. The experiment logged mouse actions to obtain accuracy and performance data on the number entry tasks.

2.3 Design

The experiment was a within-subject repeated measures design. Each participant used both number entry interfaces. The number entry interface style was the independent variable and it had two levels: the incremental and serial interfaces. The order in

which the interfaces were tested was counterbalanced for all participants. The dependent variables were the number of undetected errors, number of corrected errors, total eye fixation time on the input and display part of the interface and task completion times.

2.4 Procedure

All participants were tested individually. The eye tracker was calibrated for each participant and they were briefed about the stages and purpose of the experiment before starting.

The experiment itself was in two parts: one for each interface. Prior to each part of the experiment, the participant got a training session where they could enter 10 numbers and get familiar with the interface. When the participant was comfortable with how the interface worked, they were allowed to proceed to the experiment.

For the experiment, each participant was required to enter 100 numbers using both interfaces in the order defined by the experimenter. The participants were instructed to enter the numbers as quickly and as accurately as possible. An instruction on the right half screen showed what number the participant should enter. The participant had to click a 'Next' key to confirm their entry, and display the next instruction. The process of number entry and confirmation of entry was repeated until all 100 numbers had been entered using the first interface. The participant was allowed a break of up to 5 minutes before proceeding to the second half of the experiment. The interface was then switched and the participant went through the training session for that interface and proceeded to enter the same set of 100 numbers.

2.5 Results

Four participants were excluded from the statistical analyses. One participant's error rate was more than two standard deviations from the mean of error rates, and three had very low eye tracking data. Undetected errors were instances of number entry errors that were not caught and corrected by the participants before confirming the entry. These were not limited to the four key bounce errors introduced in the experiment. All but two participants had at least one undetected error in the experiment. To check for learning effects on the interfaces over the experiments, we analysed the task completion times from each interface in blocks of 10 trials per participant and we found no significant difference in the mean time per block.

Corrected errors for each participant on the serial interface were calculated as the total number of times they pressed the 'Cancel' button. For the incremental interface, the corrected errors for each participant were calculated as the number of times the participant overshot or undershot the target number. In the incremental interface, overshooting the target number was sometimes intentional especially when entering numbers efficiently using a mixture of continuous and discrete actions. For instance, entering the value '5.9' efficiently means slightly overshooting the target number using the continuous (hold-down) interaction in order to reach '6', for instance, and

refining the value with a down click (a discrete action) to obtain the target number. This type of intentional overshooting did not count as a corrected error.

Undetected Errors

Analysis of the total undetected errors for both interfaces using a paired *t*-test indicated that the mean undetected errors for the incremental interface (mean=1.56, sd=1.76) was significantly lower than that of the serial interface (mean=3.61, sd=2.38), $t(17)=-4.15$, $p<0.001$.

The total undetected forced errors per participant on the incremental interface (mean=0.28, sd=0.46) was significantly lower than that of the serial interface (mean=1.72, sd=1.23), $t(17)=4.74$, $p<0.001$.

Corrected Errors

The number of corrected errors per participant on the incremental interface (mean=74, sd=17.31) was significantly greater than the number of corrected errors in the serial interface (mean=7.1, sd=9.6), $t(17)=17.22$, $p<0.001$.

Visual Attention

The total visual fixation duration on the input of the serial interface (mean=271.16secs, sd=80.01), was significantly greater than the total fixation duration on the display of the device (mean=26.28secs, sd=19.31), $t(17)=13.35$, $p<0.001$. Conversely, the total fixation duration on the input of the incremental interface (mean=185.82secs, sd=87.78) was significantly lower than the fixation duration on the display (mean=553.47secs, sd=276.25), $t(17)=7.34$, $p<0.001$.

Speed of entry

The speed of entry per trial for the incremental interface (mean=8.2secs, sd=2.32) was significantly slower than the speed of entry per trial for the serial interface (mean=1.65secs, sd=0.33), $t(17)=12.71$, $p< 0.001$.

Error Types

Keystroke logs from all participants were analysed for this section. Below, we report a selection of the undetected error types that occurred in our experiment. Wiseman, Cairns and Cox have developed a taxonomy of number entry errors [6] and have independently reported and classified these errors. As well as reporting error types, we report the prevalence of certain error types between the two number entry interface styles. The frequency of each error type is shown in Figure 2. For each error type we quantify the severity of the errors by reporting the mean and standard deviation of the difference between the intended number and the transcribed number.

Missing Decimal Point Errors

This error occurs when a decimal point is absent from the transcribed number but is present in the instruction. There were 28 instances of this error on the serial interface and none on the incremental interface. On average, this error changed the intended number by 260.77 (sd=240.85).

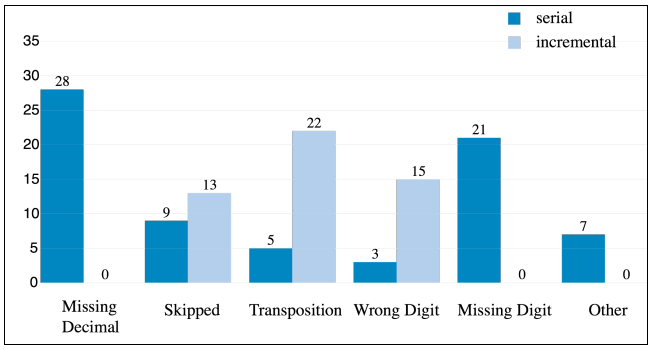

Fig. 2. The distribution of undetected error types in each interface in the experiment. Skipped errors were due to double keying errors on the "Next" key during the experiment.

Transposition Errors

Transposition errors occur when the user swaps two adjacent digits in a number. For instance instead of entering 5.84, a user might enter 5.48. The majority of these were committed on the incremental interface. The dyslexic participant committed no transposition errors. Most of the transposition errors occurred after the decimal point.

In our data, a special case of the transposition error occurred when the decimal part of the transcribed number was exactly 10 times more or less than the decimal part of the intended number. For example, instead of entering 7.4, a participant entered 7.04. Although one participant committed 17 of these errors, the potential causes make it a concern for further investigation. It is possible that the display of the numbers on the incremental interface was responsible for this error: the display always shows three significant digits. For instance if the numeric value is 7.4, the display shows 7.40. It may be confusing that the 40 after the decimal point is perceived to be greater than 4. It is important to note that this participant did not commit any transposition errors on the serial interface. It seemed that the incremental interface had an effect on their transcription of numbers specifically for numbers of the form 'd.0d' where 'd' is a numeric digit. In other words, the interface design might have affected their perception of numbers of a certain format.

Transposition errors were more serious on the incremental interface. On average this error changed the intended number by 0.54 (sd=0.35) on the incremental interface compared to 0.31 (sd=0.18) on the serial interface.

Wrong Digit Errors

Wrong digit errors occur when one of the digits in the transcribed value is incorrect. This error was more common in the incremental interface. The most serious cases of the wrong digit error happened whenever the whole number part of the number is wrong. For instance a participant entered 4.87 instead of 5.87. Wrong digit errors were more serious on the serial interface but more frequent on the incremental interface. On average, this error changed the intended number by 0.81 (sd=1.27) on the serial interface compared to 0.28 (sd=0.40) on the incremental interface.

Missing Digit Error
This refers to instances of errors where one digit from the intended value is missing from the transcribed value. For instance a participant entered 0.3 instead of 0.43. On average, this error changed the intended number by 3.36 (sd=8.92). The incremental interface was free of this error.

3 Discussion

The results show a significantly higher number of undetected errors on the serial interface in comparison to the incremental interface. This relative accuracy however comes with a slower data entry speed on the incremental interface, which may in itself account for the more accurate performance on the incremental interface. The higher level of visual attention paid to the display of the incremental interface is another possible reason for its higher accuracy. A third reason could be that participants expect to make errors on this interface. Indeed, the results show a significantly higher number of corrected errors on the incremental interface in comparison to the serial interface. Some participants had a number of tries overshooting and undershooting for the intended number before precisely setting the number. Some deliberately overshoot the intended value and correct the error in a few clicks because that is the optimal way to enter the intended number.

For the incremental interface, the visual attention placed on the input was significantly lower than that on the display. This supports our original intuition, as the interaction on the incremental interface requires the user to monitor how the value on the display changes based on what key the user is pressing. The input part of the incremental interface requires little visual attention and is only used to switch direction and precision of change. However, despite the high attention paid to the display of this interface, the mode of interaction introduced errors that were less likely on the serial interface e.g., the wrong digit errors and the transposition errors.

The results also show that the visual attention placed on the input in the serial interface was significantly higher than the visual attention on the display. This could be because participants did not feel the need to verify their entry. It is possible that most participants trusted the visual feedback they got from the labels on the keys and felt little need for an extra mode of feedback by checking the display.

By design, the numbers specified on a serial interface require parsing to obtain a numeric value valid in the application space. As a result, serial interfaces are prone to syntax errors. Rather than alert users to errors, this parsing process often produces incorrect and unpredictable results whenever the user commits a syntax error [3, 4].

Syntax errors are however impossible on an incremental interface since the application guides the user through a valid range of numeric values. It is also plausible that numbers are perceived as a string of characters when using a serial interface whereas using an incremental interface forces users to be aware of the numeric values and the relative order of numbers.

In a safety critical context like healthcare, the incremental interface is safer. It allows better error detection and the severity of errors is much lower than on the serial interface. The missing decimal point and the missing digit errors are the most serious

errors and they were both more likely to occur on the serial interface. Overall, the errors on the incremental interface had a much lower deviation from the intended number as shown by the difference of intended value to transcribed value (serial interface: mean=70.91, sd=166.94, incremental interface: mean=0.93, sd=1.41).

4 Conclusions

There are significant differences in the error rates for the two experimental conditions of number entry: number entry interface styles do affect error rates — and, by implication, medical outcomes. The speed of the serial interface comes at a price: errors are more likely to go undetected due to significantly less visual attention on the interface and undetected errors like the missing decimal or missing digit are more likely to have serious outcomes typically producing numbers out by a large factor (10 or more) from the intended values. In a medical context, such errors can be fatal.

The result suggests that it should be a priority to research number entry styles and their relation to error rates, behaviour and performance. There is a wide variety of number entry styles in medical devices (where errors cause adverse events), clearly with no or little empirical justification; we now see useful progress can be made to provide sound guidance for designers of safety critical number entry systems.

Acknowledgements
Funded as part of CHI+MED: Multidisciplinary Computer-Human Interaction research for design and safe use of interactive medical devices project EPSRC Grant Number EP/G059063/1. Duncan Brumby provided invaluable feedback on this paper.

5 References

1. ISMP: Double key bounce and double keying errors. In: ISMP Medication Safety Alert! Acute Care (January 2006). See http://www.ismp.org/newsletters/acutecare/articles/20060112.asp (Accessed April 2011)
2. Jani, Y.H., Barber, N., Wong, I.C.K.: Paediatric dosing errors before and after electronic prescribing. Quality and Safety in Health Care 19(4), 337 –340 (2010)
3. Thimbleby, H.: Interactive systems need safety locks. ITI2010: The 32nd International Conference on Information Technology Interfaces, 29–36 (2010)
4. Thimbleby, H., Cairns, P.: Reducing number entry errors: solving a widespread, serious problem. Journal of the Royal Society Interface 7(51), 1429–1439 (2010)
5. Vicente, K.J., Kada-Bekhaled, K., Hillel, G., Cassano, A., Orser, B.A.: Programming errors contribute to death from patient-controlled analgesia: case report and estimate of probability. Canadian Journal of Anesthesia 50(4), 328-332 (2003)
6. Wiseman, S., Cairns, P., Cox, A.: A Taxonomy of Number Entry Error. HCI2011: The 25th BCS Conference on Human-Computer Interaction (2011).
7. Reason, J.: Human Error. Cambridge University Press (1990)