

Text Entry Method for Reduced Keypads using One Key Stroke and One Column Stroke per Character *

Sumanta Guha

Computer Science & Information Management Program
Asian Institute of Technology
P.O. Box 4, Klong Luang
Pathumthani 12120
Thailand
guha@ait.ac.th
<http://www.cs.ait.ac.th/~guha>

Abstract. Proposed is a new method to enter text using a reduced keypad, as is typically found on telephones, with particular extensions for one with a display window above it, as is typically found on mobile phones. In this method each of the most commonly used characters is entered by selecting two keys in succession. The so-called KC-method may be programmed into a conventional mobile phone without any modification of hardware. The superiority of KC over traditional multi-tap lies in the elimination of key tap time-out, uniform text entry rate and appealing visual feedback during the entry process. KC has a patent application pending.

Key words: Mobile computing, mobile text entry, multi-tap text entry, reduced keypad, text entry, two-key text entry, two-tap text entry.

1 Introduction

Proposed is a new method to enter text using a reduced keypad, as is typically found on telephones, with particular extensions for one with a display window above it, as is typically found on mobile phones. In this method each of the most commonly used characters is entered by selecting two keys in succession – first a unique key associated to that character, followed by *any one key in a unique column of keys* associated to that character.

The so-called *KC-method* * (K for key, C for Column) may be programmed into a conventional mobile phone without any modification of hardware.

The enormous popularity of text messaging systems, particularly messages originating from mobile phones (so-called SMS messages), has created a demand for efficient methods to enter text using the reduced 12-key keypad characteristic of telephones.

* The author has a patent application pending on the method.

The most common method, implemented in almost all mobile phones, is *multi-tap* where each character is entered by pressing a particular key some number of times. Multi-tap is a *direct-entry* method in that the process of specifying each character is user-dependent from start to finish with no assistance from supporting programs. On the other hand, available nowadays with several mobile phones on the market are *knowledge-based* methods that assist the user by accessing linguistic databases. Examples are T9 from Tegic Communications and WordWise and LetterWise from Eatoni Ergonomics.

Direct-entry methods, other than multi-tap, that have appeared in published literature include *two-key methods* in which each character is specified by selecting two keys in succession. Examples are from Burrell [1] and Kandogan and Zhai [2]. As well, see discussions in the survey article by MacKenzie and Soukoreff [3]. All these two-key methods are distinguished by requiring the user to enter a character by pressing a *specific* first key followed by a *specific* second key.

KC is a two-key text-entry method as well. However, the crucial distinction from prior two-key methods is that, for most common characters, KC allows the second key press to be *any one of a set of keys* – in particular, a set which is logically and ergonomically located on the keypad. This allows not only for rapid text entry, but at a uniform rate without the key tap time-out pressure of multi-tap. Another advantage of KC is that it allows for visually appealing feedback to user key strokes, not possible with multi-tap.

2 The KC-Method

2.1 Core Implementation on a Reduced Keypad

Figure 1 shows a typical reduced phone keypad with twelve keys, arranged in three columns of four keys each.

In the core implementation of the KC-method each of the following 39 characters may be entered by pressing *two keys* in succession on the 12-key keypad: the 26 characters A-Z in the English alphabet, the 10 numerals 0-9, the space character, denoted *sp*, and the 2 special characters * and #.

The scheme for which two keys to press in order to enter a character is based on the grouping of the characters on the keys and the position of each character in its particular group. It is best motivated by example.

Consider Key 8 on the keypad. The group of three characters associated with this key are T, U and V, in that order from left to right in a row on the key itself. Accordingly, entering T requires *first* pressing Key 8 and *next* pressing *any one* key in Column 1 (i.e., *any one of Key 1 or Key 4 or Key 7 or Key **) – the choice being made of Column 1 to correspond to T because T is located leftmost in its group's row. Applying similar logic, entering U requires first pressing Key 8 and next pressing any one key in Column 2, and entering V requires first pressing Key 8 and next pressing any one key in Column 3.

Except for S and Z, other characters in the English alphabet are entered in an analogous manner: the first press is the *exact* key on which the character is

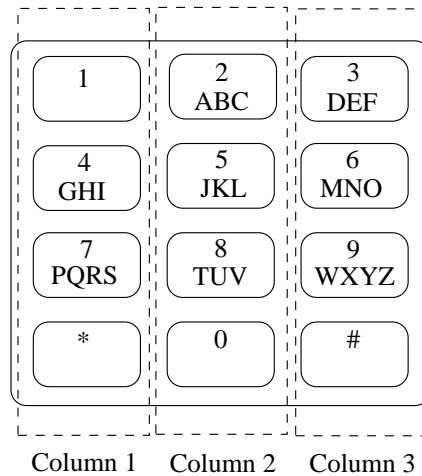


Fig. 1. Reduced keypad with the 3 columns of keys indicated.

located, and the next press is *any one* key in the column corresponding to the the character's position in its group.

The characters S and Z must be treated exceptionally as each is fourth in its respective group and there are only three columns of keys. Accordingly, KC specially associates Key # with the three characters S, *sp*, and Z in that order left to right. Therefore, entering 'S' requires first pressing Key # and next pressing any one key in Column 1, and entering 'Z' requires first pressing Key # and next pressing any one key in Column 3.

The character *sp*, which inserts a blank space in the text and is statistically the most frequent character in text messages, is treated differentially as well. It may be entered either by first pressing Key # and next pressing any one key in Column 2, or by pressing Key 0 *only once* – this is the single exception to the rule that entering a character *requires* pressing two keys in succession.

The numerals and the two special characters * and # are entered in the following manner: *first* Key * is pressed and *next* the *exact* key labeled by the desired character. E.g., '1' is entered by first pressing Key * and next Key 1.

Simple Extension: The set of characters that can be entered may be extended keeping with the spirit of the method. For example, the core implementation makes no use of Key 1 as a first press. Accordingly, a first press of Key 1 may be applied to change mode, say to lower case, or to allow other special characters, such as &, to be entered depending on the next key pressed. Implementations of the KC-method in mobile phones, in particular, may be extended by taking advantage of other keys normally present, in addition to the 12 character keys.

2.2 Extension to a Device with a Display Window Above the Keypad

We next describe particular extensions to enhance the implementation of the KC-method in a phone with a display window, as is characteristic of mobile units, based on *visual cueing*. The possibility to do this marks a significant advantage of KC over multi-tap

Cueing the User After the First Key Press Through the text entry process, when the user presses any of Keys 2, 3, 4, 5, 6, 7, 8, 9, or # as the *first* key press, the available three character selections for the next key press are displayed in a row within and at the top of the display window, in a rectangular so-called Selections Display Area (SDA). Beneath the SDA is the area of the display window where the message being entered is shown.

Moreover, each of the three character that appears in the SDA is positioned vertically above the column of keys that selects it in order to *visually cue* the user to the correct column for the next press. E.g., a first key press of Key 8 will cause the characters T, U and V to appear in a row in the SDA positioned, respectively, above Column 1, Column 2 and Column 3. This case is illustrated in Figure 2(a) where the user, in the process of entering HELLO TOM, is about to key in the first letter of TOM.

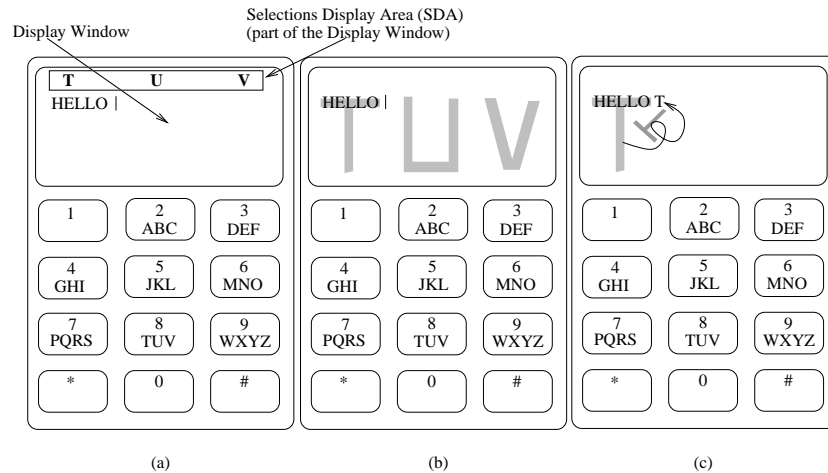


Fig. 2. (a) Visual cueing after the first press: the SDA after pressing Key 8 (b) A whole-window SDA blended in (c) Visual cueing after the second press.

In the particular case of Key # as the first press, the SDA shows S, the text string “sp”, and Z in a row. We omit details of the visual cueing after a first press of Key * or Key 0.

Remark 1. If the graphics allows the SDA could occupy the whole window with the selections *blended in* using a low alpha value (e.g., see Figure 2(b)).

Cueing the User After the Second Key Press As described previously, if any of Keys 2, 3, 4, 5, 6, 7, 8, 9, or # has been pressed first enter a character, then the SDA displays the available character selections in a manner as to cue the user to the correct column for the second press. When the selection is, in fact, completed with the next key press, the two characters not selected disappear instantaneously from the SDA while the selected character remains for a fraction of a second before disappearing as well to leave the SDA empty – the effect is of this character flashing to cue the user to its selection. At the same time, of course, the selected character appears at the end of the message being entered. We omit the remaining details of visual cueing after a second press.

Remark 2. More complex and appealing transitions from the SDA to the message may be envisaged, e.g, some kind of quirky animation (e.g., see Figure 2(c)). This will lend a video-game-like feel to the text entry process.

3 Implementation on a Mobile Phone and Conclusions

We have coded a “no-frills” implementation of KC as a Java application on a Nokia 6610 mobile phone. Experimental comparisons with multi-tap indicate that:

1. Peak rate of text entry is approximately the same with both KC and multi-tap.
2. User satisfaction is significantly higher for KC than multi-tap because of:
 - (a) Uniform rate (chosen by the user) for key tapping in KC vs. the varying number of taps per character in multi-tap.
 - (b) No time-out pressure for KC vs. having to tap successively within the machine-set time-out period of multi-tap to prevent locking in the wrong character.
 - (c) Constant visual feedback from the message area in KC reduces the tedium of text entry compared to multi-tap.

We conclude that KC is an attractive and economical alternative to multi-tap on mobile phones. Please contact the author for patent-related issues.

Acknowledgment. We thank Vu Dinh Van for implementing KC in Java.

References

1. James W. Burrell IV, Method of using a nine key alphanumeric binary keyboard combined with a three key binary control keyboard, *US Patent 6043761*.
2. E. Kandogan, S. Zhai, Apparatus and method using color-coded or pattern-coded keys in two-key input per character text entry, *US Patent 7362243 assigned to IBM Corporation*.
3. I. S. MacKenzie, R. W. Soukoreff, Text Entry for Mobile Computing: Models and Methods, Theory and Practice, *Human Computer Interaction* **17** (2002), 147-198.