Alphabetically Constrained Keypad Designs for Text Entry on Mobile Devices

Jun Gong and Peter Tarasewich
HCI Laboratory, College of Computer & Information Science, Northeastern University
360 Huntington Avenue, 202 WVH, Boston, MA 02115 USA
{gjoliver, tarase}@ccs.neu.edu

ABSTRACT
The creation of text will remain a necessary part of human-computer interaction with mobile devices, even as they continue to shrink in size. On mobile phones, text is often entered using keypads and predictive text entry techniques, which attempt to minimize the effort (e.g., number of key presses) needed to enter words. This research presents results from the design and testing of alphabetically-constrained keypads, optimized on various word lists, for predictive text entry on mobile devices. Complete enumeration and Genetic Algorithm-based heuristics were used to find keypad designs based on different numbers of keys. Results show that alphabetically-constrained designs can be found that are close to unconstrained designs in terms of performance. User testing supports the hypothesis that novice ease of learning, usability, and performance is greater for constrained designs when compared to unconstrained designs. The effect of different word lists on keypad design and performance is also discussed.

Author Keywords
Mobile device user interface design, predictive keypad text entry, novice learning and usability.

ACM Classification Keywords
H5.2. Information interfaces and presentation (e.g., HCI): User Interfaces.

INTRODUCTION

Mobile devices play an increasing role in supporting the interactions of our society. While pictures and sounds are routinely sent and received with these devices, they continue to process a great deal information in the form of text. Therefore, text entry will remain a necessary part of human-computer interaction with mobile devices, even as they keep shrinking in size.

While mobile devices may not be suitable for writing novels, they are appropriate for writing notes and "to do" lists. Text entry is also used to maintain mobile calendar and address book applications, and mobile communication needs include text messaging, retrieving information, and entering phone numbers. But mobile applications require efficient ways to record and access information under circumstances that are often quite different from those where desktop computers are used [2]. Since mobile devices can be taken anywhere, the user's environment can change rapidly from moment to moment. There can also be a significant number of people, objects, and activities vying for a user's attention aside from the mobile application itself [22]. Sometimes mobile users will find themselves wanting to enter text with one hand, or even no hands, and may not be able to look at the device itself while doing so.

Predictive text entry methods are often used with keypads on devices such as mobile phones. Past research [e.g., 9, 11, 12, 13] has looked at optimizing text entry performance by creating keypad designs (mappings of letters to keys) that reduce the number of keystrokes needed to enter a word, or reduce the number of ambiguous keystroke combinations (i.e., keystroke sequences that correspond to multiple possible words) This research looks at developing predictive text entry methods with keypads in a number of sizes (i.e., total number of keys) with the constraint that letters placed on keys must remain in alphabetical order across keys. It is hypothesized that this constraint will increase ease of learning, performance, and usability for novice users.

In addition, the more casual, unstructured, and hurried text entered into mobile devices can be very different than that used in more formal writing or in speech. A great deal of mobile communication is done with short messages, oftentimes using abbreviations such as HRU for "how are you" [1]. If predictive text entry methods use dictionaries that do not contain appropriate vocabulary, text entry will become more difficult for the user rather than easier. Text entry methods must be developed with words and phrases suited to the intended application environment. To address this concern, our research uses spoken and written word lists, as well as a list from derived from SMS (Short Message Service) messages.

In this study, complete enumeration is used to find keypad designs with alphabetic constraints for which users will have the best chance of inputting words in the fewest keystrokes without running into ambiguity problems. These designs are compared to unconstrained designs (found using a Genetic Algorithm-based heuristic) that do not require letters to remain alphabetized. Performance predictions with written, spoken, and SMS word lists show that constrained designs come very close to unconstrained designs with the same number of keys. User testing comparing an eight-key constrained keypad design against a size-equivalent (i.e., same number of keys) unconstrained design supports the hypothesis that novice user testing and usability is greater for constrained keypad designs versus unconstrained designs. Differences in keypad designs and performance based on the word list used are also discussed.

BACKGROUND

Mobile phones keypads follow an international standard where the keys 2 through 9 are each mapped to three or four letters [3]. Several approaches are used to enter text with this keypad. In the *multi-press* (or *multi-tap*) method, the user presses the key that corresponds to a desired letter one or more times. For example, the letter "r" requires that the 7 key (labeled "pqrs") be pressed three times. A user pauses or presses an additional key to move to the next letter. The *two-key* approach requires selecting a letter's group with a first key press and location with a second. With this method, the letter "r" requires the key press sequence 7-3.

A third approach uses single key presses and linguistic knowledge to "guess" the intended word. For example, the sequence 8-4-3 (corresponding to "tuv-ghi-def") might produce "the" out of all possible letter combinations. This process of *disambiguation* must sometimes automatically choose between more than one word produced by the same set of key presses, or present the user with a series of word choices. For example, the sequence 3-6-4 ("def-mno-ghi") might mean dog or fog. Word selection can be done through a combination of word or n-gram (sequence of *n* characters) frequency lists, user preferences, or past history of user word choices. A common disambiguation text entry system is T9 by Tegic (www.tegic.com).

Models for text entry using keypads [8, 21] have predicted expert input speeds as high as 27 words per minute (wpm) with multi-tap and two-key methods, and 46 wpm for predictive methods. However, actual user studies have shown lower performance. An experiment using a mobile phone [8] found that experts and novices reached about 8 wpm with multi-tap. In comparison, the T9 method was used by novices at 9.1 wpm and by experts at 20.4 wpm. In an effort to more accurately predict novice user performance, Pavlovych and Stuerzlinger recently proposed various models for non-expert keypad text entry [18]. Variations of keypad text entry techniques have looked at improving performance [17] and extending character sets beyond letters [7]. A novel "one-press" method for a mobile phone [23] even used tilting after pressing a numeric key to select a letter on the key.

Research on text entry disambiguation and keypad design can be traced back to work that originally looked at ways help motor-impaired users (those have difficulty moving their hands and arms) enter text. Research such as [9], [11], [12], and [13] described the optimization of character placement on various-sized keypads and the development of disambiguation algorithms using word dictionaries and statistics. Other research has looked at improving the performance of disambiguation algorithms by matching unknown words with the prefix or suffix of known words [19], using the context of the phrase or document being typed [16], or using probabilistic data from word pairs to guess a word based on its preceding word [4]. MacKenzie et al. developed a method that uses a stored database of prefix probabilities to guess the most likely letter of a key press based on the previously typed letters [14]. Since the method does not use a dictionary, it works well even with proper nouns, abbreviations, or slang.

PREDICTIVE KEYPAD DESIGN PROBLEM

The *predictive keypad design problem* places *M* letters on *N* keys to maximize (or minimize) a performance objective *P*. In this research, two different metrics for *P* were investigated. One metric was the *disambiguation accuracy* (DA), which indicates the fraction of times in which the word with the highest frequency of occurrence is the one intended by the user. This is the probability that, if a sequence of keys is pressed, the "desired" word appears. An optimal design is one that maximizes the DA for a given list of words. The second metric is the average number of keystrokes necessary to enter each character, or keystrokes per character (KSPC). This assumes that the user must sometimes make additional key presses to select between word possibilities resulting from ambiguous key patterns. KSPC is minimized for best performance.

The predictive keypad design problem is difficult because if more than one letter can be placed on a given key, then a given key sequence might correspond with many possible words. Shifting letters between keys can reduce the possibility of this happening. The *unconstrained* version of the keypad problem allows any letters to be placed on any keys, and the *constrained* version requires letters to remain in alphabetical order across all keys. Previous studies have shown that users of predictive keypad methods with unconstrained letter placement can achieve high performance, but only after much practice [8]. Alternatively, constrained designs should increase usability and lessen learning time for novices. This hypothesis seems reasonable given the work of Smith and Zhai, who found that a virtual keyboard created with alphabetical ordering tendency (i.e., keeping letters *close* to alphabetical order), while not optimal for advanced users, did increase performance and preference ratings for novice users [24]. In

addition, alphabetization should allow novices to build expertise more quickly.

Word prediction algorithms match the current letters entered by a user to a dictionary of words with associated frequencies of occurrence. If the current letters (determined from key presses) match an entire word, that word becomes a candidate for selection. The frequencies determine which candidate ranks highest on a selection list (and appears on the screen). The user can accept that word choice, or cycle through any remaining choices on the list (usually by pressing a key labeled "Next").

FINDING KEYPAD DESIGN SOLUTIONS

The problem of arranging of letters on keys belongs to the NP-complete class of mathematical optimization problems [11]. Therefore, the most straightforward way to guarantee finding the optimal solution is by searching every possible arrangement and selecting the best one. Enumerating the total number of different unconstrained keypad arrangements quickly becomes computationally unrealistic as the number of letters and keys increases. Fortunately, it is easier to completely enumerate all of the constrained layouts. The constrained problem can be viewed as placing N-1 dividing positions between a sequence of M characters, which can be solved completely in a more reasonable time. For example, for M=26 and N=8 (the number of keys with letters on a standard mobile phone keypad), the unconstrained problem has $\sim 1.6 \times 10^{20}$ possible solutions, while the constrained design has only 480,700. While large problems still took several days to solve, complete enumeration was used to calculate all possible constrained keypad designs for varying numbers of keys. A program simply generated all of the character/divider sequences, translated each into the corresponding keypad design, checked its performance, and tracked the best one.

For comparison purposes, unconstrained keypad designs were generated using a Genetic Algorithm-based heuristic. Genetic Algorithms (GA) simulate biological evolution behaviors such as reproduction, crossover, and mutation to quickly search for solutions to complex problems [5]. Our method first randomly generates a population of candidate solutions. The best solutions from this population "mate" with each other, and their "children" become the next generation of solutions. Reproduced children are also subject to random changes ("mutations"). The best solutions of this next generation are then used to reproduce another generation. This process continues for a predetermined number of iterations. Additional details on the GA-based heuristic are available from the authors. The heuristic performed quite well when its results for constrained keypad designs were compared against results found by complete enumeration. The heuristic often found the optimal solution, and always came within 0.33% of the optimal solution for one to twelve key designs. Given this, it seems reasonable that the unconstrained designs found using the heuristic, if not optimal, will be quite close to optimal (although not necessarily within 0.33% of optimal, since that result applies only to constrained designs).

CORPORA

One limitation of most text entry research for mobile devices is that frequency information for words and phrases comes from text corpora derived from publications such as newspapers and books. But text entry in the mobile environment can be quite different from that in the desktop environment in terms of words used and their frequency of usage. Text entry can also vary based on the context of use, and the task being performed. To see what effect different task environments might have on text entry interface design, we used three distinct English word lists in this study. One list was derived from written language, one from spoken language, and one from short text messages.

The written and spoken word lists[1] used in our study are derived from the British National Corpus (BNC) [10], which contains approximately 65,000 distinct words and their frequencies of occurrence. These words are derived from a total sample of 100 million words of present-day English, of which approximately 90 million come from books and ten million come from conversations and monologues. The written word list, with a total of 4420 words, contains all words that appear with a frequency of at least 20 times per million written words. The spoken word list, with a total of 4307 words, contains all words that appear with a frequency of at least ten per million spoken words. While these two lists are shorter than the complete word lists derived from the BNC, the written word list used in our study accounts for approximately 84% of all written words, while the spoken list accounts for about 96% of all spoken words. The written word list is similar to those used in many previous mobile device text entry studies. The spoken word list should better reflect the mobile environment in terms of word usage and frequencies because speech is usually less formal than writing.

The SMS word list, which contains 7325 words, was derived from a corpus[2] consisting of about 10,000 SMS messages collected as part of a research study in Singapore [6]. The study looked at ways to improve SMS messaging on mobile phones by remapping characters to keys (but without alphabetization constraints). Two-thirds of the SMS corpus was collected from several regular users of mobile phones. The rest was primarily collected over the Web from a larger sample of other solicited mobile phone users. The corpus was processed to create a word list and a set of associated frequencies.

There are several differences that stand out when comparing the spoken, written, and SMS word lists.

[1] Written and spoken word lists are available at: www.comp.lancs.ac.uk/ucrel/bncfreq

[2] The SMS corpus is available at: www.comp.nus.edu.sg/~rpnlpir/downloads/corpora/smsCorpus/

Abbreviations, such as "hrs" for "hours", and "vry" for "very," occur frequently in the SMS corpus. Abbreviations containing periods (e.g., "U.S.A."), and large numbers (e.g., "12,345"), occur only in written language. On the other hand, single letters, such as "U," "S," and "A," tend to occur only in speech [2]. Significant differences in word frequencies also occur between the three lists. For example, the word "you" occurs 25,957 times in one million spoken words, about 9700 times in one million SMS words, and only 4,755 times in that many written words. The word "yeah" occurs 7,890 times in speech and only 17 times in written text per one million words.

Keys	Written	Spoken	SMS
1	17.29%	14.60%	7.79%
2	50.72%	45.77%	31.42%
3	74.38%	69.45%	54.36%
4	85.35%	79.89%	68.71%
5	91.10%	87.25%	78.13%
6	93.85%	91.97%	84.84%
7	95.96%	95.04%	89.55%
8	97.11%	96.69%	92.49%
9	97.70%	97.72%	94.02%
10	98.13%	98.41%	95.13%
11	98.51%	99.01%	96.11%
12	98.76%	99.32%	96.85%

Table 1. Maximum disambiguation accuracy values for 1 to 12 key constrained keypad designs by word list

Written	Spoken	SMS
ABCDEF	ABCDEF	ABCDEFG
GHIJKLMN	GHIJKL	HIJKLM
OPQRS	MNOPQ	NOPQRS
TUVWXYZ	RSTUVWXYZ	TUVWXYZ
ABCD EFGH	ABCD EFG	ABCD EFG
IJKL MN	HIJKL MN	HIJ KLM
OPQR S	O PQRS	NOPQ RS
T UVWXYZ	TUV WXYZ	T UVWXYZ
ABCD EF	ABC DEF	ABCD EFG
GH IJKL	GH IJKL	HIJ KLM
MN OPQR S	MN O PQRS	N OPQ RS
T UVWXYZ	TUV WXYZ	T UVWXYZ
AB CD EF	AB CD EF	AB CD EFG
GH IJKL M	GH IJKL M	HIJ KL M
N O PQR	N O PQR	N OP QRS
S TUV WXYZ	S TUV WXYZ	T UVW XYZ

Table 2. Selected optimal constrained keypad designs for maximum DA values by word list (4, 8, 9 and 12 keys)

Two modifications were made to all three lists before they were used in our study. First, all punctuation was removed and any upper-case characters were converted into lower-case. So occurrences such as "let's" and "U.S.A" became "lets" and "usa". This was done because the study focused only on creating a keypad for only the letters of the English alphabet, without punctuation marks or capitalization. This assumption is limiting, but consistent with other studies of text entry methods. Second, any phrase found on the list was removed, and its frequency of occurrence was added to the frequencies of all its member words. For example, if the phrase "in addition to" appeared ten times, ten was added to the frequencies of "in", "addition", and "to". This modification assumed that any dictionary used for prediction would consist of words and not phrases.

KEYPAD DESIGN RESULTS

Using complete enumeration, optimal keypad designs were found for keypad sizes ranging from one to twelve keys. Designs were optimized by maximizing disambiguation accuracy using each of the three word lists (values shown in Table 1). Selected optimal designs (for four, eight, nine, and twelve keys) for all three lists are shown in Table 2.

Complete enumeration was then used to find optimal keypad designs using the KSPC metric. Table 3 shows the minimum KSPC values found for keypad designs having one to twelve keys across the three different word lists. Table 4 shows selected optimal designs corresponding to these KSPC values.

Keys	Written	Spoken	SMS
1	67.85	87.94	211.28
2	4.62	6.64	20.24
3	1.99	2.66	6.92
4	1.45	1.71	3.69
5	1.25	1.41	2.54
6	1.14	1.26	1.92
7	1.10	1.17	1.63
8	1.07	1.12	1.43
9	1.06	1.08	1.31
10	1.05	1.05	1.25
11	1.05	1.03	1.20
12	1.04	1.02	1.16

Table 3. Minimum KSPC values for 1 to 12 key constrained keypad designs by word list

Written	Spoken	SMS
ABCD	ABCDEF	ABCDEF
EFGHIJKL	GHIJKLM	GHIJKLM
MNOPQRS	NOPQRS	NOPQRS
TUVWXYZ	TUVWXYZ	TUVWXYZ
AB CDEFG	ABC DEFG	ABCD EFG
HIJKL MN	HI JKLM	HIJ KLM
OP PRS	NOPQ RS	NO PQRS
T UVWXYZ	TU VWXYZ	TU VWXYZ
AB CDE	AB CDE FGH	AB CDE
FG HIJKL	I JKLM	FGH IJK
MN OPQR S	NOPQ RS	LM NO PQRS
T UVWXYZ	TU VWXYZ	TU VWXYZ
A BCD EFG	A BCD EFG	AB CD EF
H I JKL M	H I JKL	GH I JKL M
NO PQR S	M NO PQR	N O PQRS
T UVWXYZ	S TUV WXYZ	TU VWXYZ

Table 4. Selected optimal constrained keypad designs for minimum KSPC values by word list (4, 8, 9 and 12 keys)

Since the search space for unconstrained keypad designs is so large, it is impractical to use complete enumeration to find optimal layouts. Instead, our GA-based heuristic was used to find solutions to this problem. Maximum disambiguation accuracy values for each word list are shown in Table 5. Selected optimal keypad designs are shown in Table 6.

Keys	Written	Spoken	SMS
1	17.29%	14.60%	7.79%
2	52.39%	48.67%	33.61%
3	69.69%	72.31%	56.36%
4	87.39%	85.27%	70.45%
5	90.17%	87.02%	70.45%
6	96.10%	96.40%	80.82%
7	97.20%	97.72%	88.28%
8	98.05%	98.50%	92.43%
9	98.55%	98.81%	94.31%
10	98.78%	99.27%	96.42%
11	98.84%	99.42%	97.34%
12	99.04%	99.51%	97.43%

Table 5. Maximum DA values (using the GA-based heuristic) for 1 to 12 key unconstrained keypad designs by word list

Written	Spoken	SMS
ABCDFT	AFTUVXY	ANPT
EILWY	BCDEL	BFIKSY
GKMNOQVZ	GHKMOS	CEJLMUXZ
HJPRSUX	IJNPQRWZ	DGHOGRVW
AQT ES	AFLX MOQ	AWY ESX
BNZ FHJOX	BIT HS	BNZ FGPQU
CIKY GPR	GPUY DEV	CLRV IJT
DUVW LM	NWZ CJKR	DHK MO
ACJK BDF	AFP HIKQYZ	AD BGR
EGM HOX	CN DR	CJOZ EHX
ITV LQY NW	CM LO EW	NW IT KSV
PRZ SU	JSVX BTU	LMQ FPUY
AFP BG CE	DU EW KR	AH BFZ CR
DHQ IJKYZ LV	HP CL S	DQU EJV GM
M N OS	IJQVX H BY	IY KLP N
R T UWX	AT GM FOZ	OW SX T

Table 6. Selected optimal unconstrained keypad designs for maximum DA values by word list (4, 8, 9 and 12 keys)

Keypad Designs	1	2 ABC	3 DEF	1 AV	2 EBJ	3 IRZ	1 ABX	2 DEF	3 GHI	1 ALV	2 EDG	3 IMK
	4 GHI	5 JKL	6 MNO	4 OMP	5 HUY	6 TXQ	4 OKL	5 MNJ	6 PQR	4 OP	5 SUX	6 TCW
	7 PQRS	8 TUV	9 WXYZ	7 NKW	8 SFG	9 LDC	7 SU	8 VWC	9 YZT	7 NYQ	8 RFJ	9 HBZ
Name	North America Standard			Levine			TOC			Frequency		
Disambiguation Accuracy (DA)	95.41%			97.11%			96.63%			93.52%		
Comparison Design	8-Key Constrained			9-Key Constrained			9-Key Constrained			9-Key Constrained		
Comparison DA	96.69%			97.72%			97.72%			97.72%		
Improvement	1.34%			0.61%			1.09%			4.20%		

Table 7. Comparison of constrained keypad designs with other keypad designs (spoken wordlist)

Constrained keypad designs optimized on written word list	ABCDEF	ABCD	EFGH	ABCD	EF		AB	CD	EF		
	GHIJKLMN	IJKL	MN	GH	IJKL		GH	IJKL	M		
	OPQRS	OPQR	S	MN	OPQR	S		O	PQR		
	TUVWXYZ	T	UVWXYZ	T	UVWXYZ		S	TUV	WXYZ		
DA value if SMS word list used	67.46%	91.18%		93.01%			96.54%				
Constrained keypad designs optimized on spoken word list	ABCDEF	ABCD	EFG	ABC	DEF		AB	CD	EF		
	GHIJKL	HIJKL	MN	GH	IJKL		GH	IJKL	M		
	MNOPQ	O	PQRS	MN	O	PQRS		O	PQR		
	RSTUVWXYZ	TUV	WXYZ	TUV	WXYZ		S	TUV	WXYZ		
DA value if SMS word list used	67.50%	91.07%		92.77%			96.54%				
DA value of keypad design optimized on SMS word list	68.71%	92.49%		94.02%			96.85%				
Increase in DA over: **written list optimization**	1.85%	1.44%		1.09%			0.32%				
spoken list optimization	1.79%	1.56%		1.35%			0.32%				

Table 8. Performance comparison of keypad designs optimized on one word list, but used with another

DISCUSSION OF KEYPAD DESIGN RESULTS

The disambiguation accuracy values from Table 1 for each of the three word lists are plotted against the number of keypad keys in Figure 1. Predicted performance is poor for small designs, but increases rapidly as the number of keys increases and then begins to level off. For keypad sizes up to nine keys, the performance level for the written word list is always greater than or equal to the spoken list. For ten keys and above, the performance level for the spoken word list becomes higher than that of the written word list. Performance with the SMS word list is consistently lower than that obtained with the other two lists, and the rate of increase in performance is also lower. Very good performance (DA over 95%) was achieved with designs of seven keys for the written and spoken word lists, but not until ten keys with the SMS list. Good results (DA over

90%) were also obtained for designs with five or six keys with the written and spoken word lists, suggesting that using fewer keys (i.e., smaller keypads) may be feasible on devices were space is at a premium. But this result is tempered by results with the SMS word list, which requires seven keys to approach the same performance level. Table 2 shows that the optimal keypad designs are different for each of the three word lists, except at twelve keys, where the written and spoken list designs are the same.

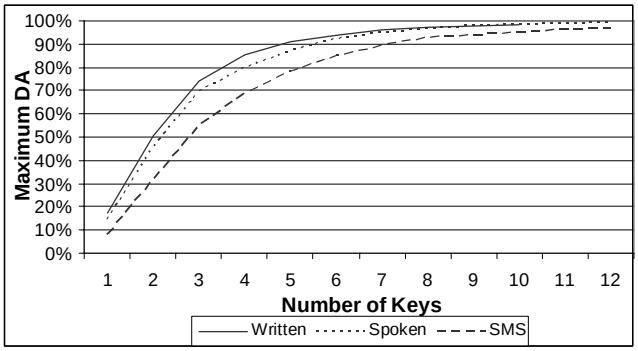

Figure 1. Maximum DA vs. number of keys for each word list (constrained keypad designs)

A similar trend in performance is found with the KSPC values in Table 3. For up to ten keys, optimum predicted performance with the written word list is greater than with the spoken word list, which in turn is greater than with the SMS word list. For eleven and twelve keys, performance becomes greatest with the spoken word list, followed by the written and SMS lists. A comparison of Tables 2 and 4 shows that for all three word lists, the two metrics provide different optimal keypad designs for a given number of keys. This suggests that optimal keypad designs might be task or condition specific, and that the choice of keypad design may depend on the type of performance that is desired. That is, whether it is more desirable to minimize the number of keystrokes, or maximize the number of correct word predictions.

Comparing Table 5 with Table 3 shows that maximum DA values for unconstrained keypad designs consistently meet or exceed the DA values of equivalent constrained designs across the three word lists. But the optimal unconstrained keypad designs (Table 5) are radically different than their constrained counterparts (Table 6). Results of a comparison test between two size-equivalent constrained and unconstrained designs are presented later in this paper.

Table 7 shows a performance comparison (using the spoken word list) between our optimal constrained keypad designs and three unconstrained designs proposed in the literature [11]. A comparison is also made to the standard North American telephone keypad, which is a constrained design [3]. Size-equivalent (that is, with the same number of keys) constrained designs from our study have performance advantages of between 0.61% and 4.20% over all four other designs. This means that for every 100 words entered on a keypad, our designs should produce approximately one to four more "correct" words than the other designs. Besides offering better performance, the constrained designs should also offer greater ease of learning and usability, especially for novices. In addition, the constrained eight-key design is similar enough to the North American design such that user effort to change over to the new design if it were implemented on mobile phones would be minimal compared to unconstrained designs.

An optimal keypad design was also proposed in the study that collected the SMS corpus used in our research [6]. The optimal constrained keypad design using the SMS word list from our study is compared to the eight-key unconstrained design from [6] in Table 9. The difference in DA values is 1.76% in favor of the SMS keypad design, but the difficulty in learning the unconstrained keypad design may negate the performance advantages that can be gained by the higher DA value.

Keypad	SMS			8-Key Constrained		
	1	2 EGP	3 CDR	1	2 ABCD	3 EFG
	4 HIVZ	5 LNQX	6 KMO	4 HIJ	5 KLM	6 NOPQ
	7 BSY	8 JTU	9 AFW	7 RS	8 T	9 UVWXYZ
DA	94.25%			92.49%		

Table 9. Comparison to optimal SMS keypad design

In addition to providing support for constrained keypad designs, our research has shown the effects of different word lists on keypad design and performance. If a design is optimized on an inappropriate corpus, performance will suffer. Table 8 shows DA values with the SMS word list for keypad designs originally created (optimized) using the written or spoken word lists. The values are consistently much lower than the DA values achieved when the keypad is designed using the SMS word list.

The effect of the word lists is also shown by the differences in optimal keyboard design (not just DA value) for different size keypads. Except for the one, two, and twelve key designs, at least one of the constrained optimal designs is different from the other two for a given keypad size. For half of the keypad sizes, all three constrained optimal designs are different.

The size of a word list can factor into how well a keypad design can be optimized in terms of DA and KSPC. The more words on a list, the harder it is to find key combinations that match only one word from the list. The SMS word list is much larger than the spoken and written word lists, so it seems reasonable that final DA and KSPC values for given keypad sizes might be lower than for the other word lists.

Another factor that can affect keypad performance is the distribution of word sizes. Table 10 shows the percentage of words of a given word length in each word list. Figure 2 summarizes this same data graphically. The SMS list

distribution is skewed leftward more than the written and spoken word list distributions. This is not surprising since it is expected that users would tend to favor shorter words for text messaging due to text entry difficulty and limitations on message size. But placing increased emphasis on shorter words, especially with larger word lists, makes it more difficult to find well-performing solutions. Distribution of word sizes may also be the reason that performance with the spoken word list becomes better than written list performance for designs of ten to twelve keys.

Word size (characters)	Percentage of Total Word List		
	Written	Spoken	SMS
2	1.95%	1.56%	4.11%
3	4.41%	5.92%	13.32%
4	13.46%	14.58%	17.39%
5	15.54%	17.48%	16.87%
6	16.02%	16.39%	15.30%
7	15.18%	15.16%	13.04%
8	11.88%	10.91%	9.01%
9	8.46%	7.64%	5.35%
10	5.95%	5.11%	2.88%
11	3.19%	2.37%	1.30%
12	1.76%	1.23%	0.67%
Other	2.19%	1.65%	0.75%

Table 10. Number of words of different lengths in different word lists

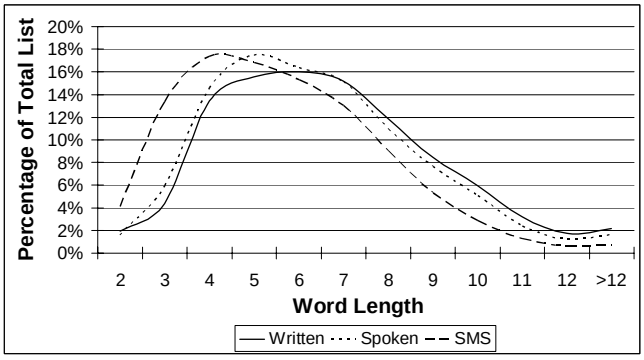

Figure 2. Total percentage of words of various lengths for each of the three word lists

TESTING THE DESIGNS

To investigate the hypothesis that the constrained keypad designs are easier for novices to learn and use than their equivalent unconstrained designs, we performed an experiment. The optimal eight-key constrained and unconstrained keypad designs found using the spoken word list were used to create two text entry interfaces using virtual keys. These interfaces were implemented on a Pocket PC using the .NET Compact Framework environment. The interface with the constrained keypad design is shown in Figure 3.

The interface was divided into two parts. The top section was a text box, used to display the text that is entered. The bottom section was a twelve-button keypad. Eight keys were assigned letters. The four remaining keys were assigned functions to create an interface that allows multiple word entry and basic editing as follows:

- **"←" key:** This "backspace" key is used to delete the last character displayed.
- **"Done" key:** This is pressed after a sentence has been entered. The text box is cleared for the next sentence.
- **"Next" key:** The "next" key is used to cycle through all the word candidates resulting from an ambiguous key press sequence.
- **"-->" key:** This key adds a space and allows the next word to be entered.

Figure 3. Constrained keypad design testing interface

Eight computer science graduate and undergraduate students voluntarily participated in the experiment. Six of the subjects were male, and two were female. The median age of the subjects was 29 years, and all but one carried a mobile phone. The other seven had carried mobile phones for 3.7 years on average, and three stated that they used the mobile phone daily. Subjects primarily used their mobile phones for voice calls, checking the time, and for reminders or alarms. Among the seven subjects who carried mobile phones, five of them reported occasional use of text messages, and the other two said they never received or sent text messages. Since none of the subjects used the phones for text messages on a regular basis, they were all categorized as novices for the purpose of this study.

Two testing sessions were conducted. For the first session, each subject first filled out a questionnaire asking for background information. The subject was then given either the constrained or unconstrained interface design (randomly

chosen). A stylus was used to "press" each key. An explanation on how to enter a sentence was given, and the subject entered two sample sentences for practice. After this training, each subject performed four task sessions. During each task session, the subject entered one of four sets of six sentences. Each subject entered all four sets over the course of the testing session, but the order of the sets was randomly chosen. A sample set of sentences is shown in Table 11. Sentences were chosen from those provided in [15]. The sets were balanced to provide the same number of characters in each set, and the same number of words that require an ambiguous keystroke sequence (and therefore use of the Next key). For each task session, subjects alternated between the constrained and unconstrained keypad designs, starting with the one they trained on, therefore using each interface twice. Opinions about the interfaces were solicited at the end of the testing session. Subjects were told to work as quickly and accurately as possible, and to correct any errors that they noticed.

an efficient way to heat a house
watch out for low flying object
pay off a mortgage for a house
express delivery is very fast
life is but a dream
have a good weekend

Table 11. Sample set of sentences

For the second session, the same subjects returned one week later to perform a similar set of tasks, although a questionnaire and training session were not administered. In addition, instead of using the unconstrained design, the constrained design was tested against the North American standard telephone keypad design. Subjects used the constrained interface for inputting the same two sentence sets they entered in session one. The remaining sets of sentences were used with the telephone keypad design.

	Constrained	Unconstrained	Sig.
Average text entry speed	6.74 WPM	5.50 WPM	0.000
Average errors per sentence	0.39	0.31	0.344

Table 12. Comparison of entry speeds and error rates for constrained and unconstrained keypad designs

EXPERIMENT RESULTS

Non-parametric (Wilcoxon) tests were used to see if there were differences in performance for the two keypad designs. For the first testing session, Table 12 shows the average input speed (in words per minute (WPM)) for the constrained and unconstrained designs, along with the average number of errors per sentence. Errors are defined as each use of the backspace key to correct a character (no other types of errors were recorded). The significance of the difference between the two values is reported in the last column. This analysis shows that average entry speed using the constrained keypad design is significantly faster than that with the unconstrained design, but the error rate is not significantly different for the two interfaces.

For the second session, Table 13 shows the average input speed for the constrained and standard telephone keypad designs, along with the average number of errors made per sentence. This analysis shows that average entry speed using the constrained keypad design is not significantly different than that with the standard telephone design, and the error rate is not significantly different either.

	Constrained	Cell phone	Sig.
Average text entry speed	7.89 WPM	8.22 WPM	0.169
Average errors per sentence	0.47	0.42	0.777

Table 13. Comparison of entry speeds and errors for constrained and standard cell phone keypad designs

Since the same sentences were input by the same subjects using the constrained keypad design in both sessions one and two, learning effects can be analyzed. Table 14 shows the average text entry speed for the constrained keypad designs in both sessions, along with the average number of errors per sentence. This analysis shows that average entry speed in session two is significantly higher than that in session one, and the error rate is not significantly different between sessions.

	Session 1	Session 2	Sig.
Average text entry speed	6.74 WPM	7.89 WPM	0.000
Average errors per Sentence	0.39	0.47	0.457

Table 14. Comparison of entry speeds and errors across two testing sessions with the constrained keypad design

DISCUSSION OF EXPERIMENT RESULTS

While this experiment is small, and only looks at three designs from one keypad size, its results generally support the hypothesis that novice user testing and usability is greater for constrained keypad designs versus unconstrained designs. From Table 12, we see that novice users were able to use the constrained eight-key keypad design more effectively than an equivalent unconstrained design. Text input speeds are higher without any difference in error rate.

Table 13 shows that while the average input speed with the standard telephone keypad is slightly higher than with the equivalent constrained design, the difference is not significant. There is no significant performance difference in terms of error rate between the two designs as well. Table 14 suggests that subjects can effectively learn and use constrained keypad techniques over time, especially since they were not provided with any additional training at the beginning of the second session.

CONCLUSION

This study has contributed to knowledge about predictive keypad text entry techniques in two ways. First, optimal

keypads of various sizes were designed under the constraint of keeping letters in alphabetical order across keys, a problem which had not been studied previously. While analysis showed that performance tradeoffs exist between the designs, reasonable performance can be achieved by relatively small keypads, and performance is close to that found for size-equivalent constrained designs. Optimal keypad designs also differed for the two performance metrics used, suggesting that the best keypad design might depend on the type of task involved. User testing of an eight-key design supported the hypothesis that novice user ease of learning and usability is greater for constrained designs than for unconstrained designs.

Second, the study showed that different word lists affect optimal predictive keypad designs. The study used three word lists from different corpora, including one created from SMS messages, and found that performance for the same size keypad can vary greatly based on the word list used. It was hypothesized that not only list size, but the distribution of word length in a list, could affect performance of keypad designs because keystroke ambiguities become more difficult to resolve. The analysis of different word lists and corpora for keypad design has not been addressed in previous studies.

A secondary contribution of this study is the development of a GA-based heuristic to find effective solutions (using reasonable computational effort) to the NP-complete problem of creating unconstrained keypad designs based on various word lists. The heuristic also finds good solutions to the constrained keypad design problem. While GA techniques have been applied to unconstrained keypad design in the past, they have not been applied to constrained keypad design.

Many "optimized" text entry methods are counterintuitive and radically different from what users are already familiar with. They can require great amounts of learning time to use effectively, and can frustrate users in the process. This study has investigated predictive keypad text entry methods that can increase ease of learning, usability, and performance for novice users. The methods conform to the basic design principle of standardization, which is done through alphabetization of letters across keys. Techniques that build on familiarity and intuition should work well not only for novices, but also for mobile users who may be attending to other tasks while entering text.

While this study makes several significant contributions, it also has limitations. First, while a full set of English letters was used for keypad design purposes, numbers and punctuation were not addressed in this study. The results may also not be directly applicable to keypad designs for languages other than English. In addition, editing was limited to a backspace command, with no ability to perform more advanced actions such as deleting an entire word. Adding such functionality would most likely require additional keys, with final keypad designs influenced by how often such functionality is used.

The written and spoken word lists used to create the keypad designs in this study were subsets of the complete lists derived from the British National Corpus. These lists were chosen to demonstrate the feasibility of designing constrained keypads and to compare their predicted performance against other keypad designs, rather than to create an "ultimate" keypad design. The words that users should need most of the time are in these lists, but less frequent words are not. Furthermore, it is not desirable to optimize keypad designs on the entire lists, because then designs will be optimized on words that may realistically never be used, which could negatively impact actual performance. Of course, it may be desirable to add some less frequently used words to the lists to minimize the occurrence of missing words. Users could also enter their own words into the system. While the original design of the keypad will not account for such additions, a frequency or ranking for the word can be entered for use during word prediction.

The SMS corpus used in this study has limitations based on its derivation and origin. While the messages were in English, they were collected from subjects in Singapore. There may be cultural or country-specific differences in the type of words or abbreviations used, or the spellings of some words. This concern comes into play with the British National Corpus as well, with the spelling of words such as color (as colour) and analyze (as analyse). The SMS corpus also consisted primarily of self-submitted messages, rather than being a true sampling of actual SMS messages sent and received by users. Certain messages may have been purposely or mistakenly left out of the corpus, and there is also the potential for fake messages to be added.

Although not a limitation, a predictive algorithm based on word matching was used for this study. However, the results of the keypad designs could differ if other methods, such as those based on n-grams or word pairs, are used. This research also did not look at how physical key configuration differences for keypad designs might affect performance. For example, comparing performance of a nine-key design arranged as a 3x3 grid or arranged as one row of four keys over another row of five keys.

The experiment reported in this paper provides a very basic level of support for the benefits of constrained keypad designs. Larger studies are needed to test a wider variety of different keypad configurations. Studies also need to be conducted to test longer-term learning effects. Realistic prototypes also need to be built to better mimic the types of mobile devices these methods are intended for.

In terms of future research, the authors plan to 1) continue user testing with various keypad configurations, 2) implement numbers, punctuation, and additional editing functions into predictive keypad design, 3) move towards mobile phones or other more realistic prototype platforms

for testing, and 4) continue investigating the effects of different word lists on keypad designs, and attempt to determine what specific aspects of word lists cause performance results to differ. It may work best to collect various corpora from different mobile environments and applications (such as text messaging, calendar functions, and information retrieval).

Another area for future research is the development and validation of user models to aid the understanding and prediction of user performance with mobile device text entry methods. Existing HCI models for keypad text entry fail on two accounts. One is in their ability to predict novice (versus expert) performance, although recent work by Pavlovych and Stuerzlinger [18] has begun to address this shortfall. To do well in predicting novice performance, models need to account for actions that may occur (or occur more often) when users are first learning a system, like pauses, double checking of actions, and wrong key presses. The other is how models account for errors, editing, and other more indirect factors that contribute to text entry performance. For example, duplicate key presses (pressing a key twice by mistake) can occur, and users might delete an entire word and type it again rather than correcting a single incorrect letter it contains. Both of these issues are critical to accurately modeling user performance, especially that of non-experts, on mobile devices. The authors have started to create models that address these two issues.

REFERENCES

1. Cricket Communications. Text message lingo. (2004). Available at: https://www.mycricket.com/lingo.aspx.
2. Gillian, H., Jeffrey, S.P., and Gregory, D.A. Practices for capturing short important thoughts, *Ext. Abstracts CHI 2003*, ACM Press (2003), 904-905.
3. Grover, D.L., King, M.T., and Kuschler, C.A. Patent No. US5818437, Reduced keyboard disambiguating computer. Tegic Communications, Inc., 1998.
4. Hasselgren, J., Montnemery, E., and Svensson, M. HMS: A predictive text entry method using bigrams. *Ext. Abstracts 10th Conf. of the European Chapter of the Association of Comp. Linguistics* (2003), 43-49.
5. Holland, J.H., Adaptation in Natural and Artificial Systems. Ann Arbor, Michigan: University of Michigan Press (1975).
6. How, Y. Analysis of SMS Efficiency. Undergraduate Thesis. National University of Singapore (2004).
7. Ingmarsson, M., Dinka, D., and Zhai, S. TNT – A numeric keypad based text input method. *Proceedings CHI 2004*, ACM Press (2004), 639-646.
8. James, C.L., and Reischel, K.M. Text input for mobile devices: Comparing model prediction to actual performance. *Proc. CHI 2001*, ACM Press (2001), 365-371.
9. Kreifeldt, J.G., Levine, S.L., and Iyengar, C. Reduced keyboard designs using disambiguation, *Proc. of the Human Factors Society 33rd Annual Meeting*, (1989), 441-444.
10. Leech, G., Rayson, P., and Wilson, A. *Word Frequencies in Written and Spoken English: Based on the British National Corpus*, Pearson ESL (2001).
11. Lesher, G.W., Moulton, B.J., and Higginbotham, D.J. Optimal character arrangements for ambiguous keyboards. *IEEE Trans. on Rehabilitation Engineering 6*, (1998), 415–423.
12. Levine, S.H., and Goodenough-Trepagnier, C. Customised text entry devices for motor-impaired users. *Applied Ergonomics 21*, 1 (1990), 55–62.
13. Levine, S.H., Goodenough-Trepagnier, C., Getschow, C.O., and Minneman, S.L. Multi-character key text entry using computer disambiguation. *Proc. RESNA 10th Annual Conference*, (1987), 177-178.
14. MacKenzie, I.S., Kober, H., Smith, D., Jones, T., and Skepner, E. LetterWise: Prefix-based disambiguation for mobile text input. *Proc. UIST 2001*, ACM Press (2001), 111-120.
15. MacKenzie, I.S., and Soukoreff, R.W. Phrase sets for evaluating text entry techniques. *Ext. Abstracts CHI 2003*, ACM Press (2003), 754-755.
16. Masui, T. POBox: An efficient text input method for handheld and ubiquitous computers. *Proc. HUC '99*, Springer-Verlag (1999), 289-300.
17. Nesbat, S.B., A fast, full-text entry system for small electronic devices. *Proc. 5th International Conference on Multimodal Interfaces*, (2003), 4-11.
18. Pavlovych, A., and Stuerzlinger, W. Model for non-expert text entry speed on 12-button phone keypads. *Proc. CHI 2004*, ACM Press (2004), 351-358.
19. Rau, H., and Skiena, S.S., Dialing for documents: An experiment in information theory. *Proc. UIST 1994*, ACM Press (1994), 147-155.
20. Sigmund, K., *Games of Life: Explorations in Ecology, Evolution and Behaviour*, Oxford University Press (1993).
21. Silfverberg, M., MacKenzie, I.S., and Korhonen, P. Predicting text entry speed on mobile phones. *Proc. CHI 2000*, ACM Press (2000), 9-16.
22. Tarasewich, P. Towards a comprehensive model of context for mobile and wireless computing. *Proc. AMCIS 2003*, AIS (2003), 114-124.
23. Wigdor, D., and Balakrishnan, R. Tilt text: Using tile for text input to mobile phones, *Proc. UIST 2003*, ACM Press (2003), 81-90.
24. Zhai, S., and Smith. B.A. Alphabetically biased virtual keyboards are easier to use - layout does matter, *Ext. Abstracts CHI 2001*, ACM Press (2001), 321-322.