A Text Input Scheme for Indic Languages with Large Numbers of Printable Characters

Abstract

This paper discusses design and development of a text-input scheme for phonetic Brahmic languages with a large number of printable characters. We devise an input scheme for an exemplar Indic language with the understanding that the findings are generalizable to other Indic languages. Our results show that a casual user is able to type at a reasonable speed with our approach.

1 Introduction

The standard keyboard that comes with computers around the world is the QWERTY keyboard. However, for a large number of languages in the world, this keyboard is not ideal. For example, the QWERTY keyboard is unsuitable for the modern descendents of the Brahmi script, used widely in countries of the Indian sub-continent and parts of East Asia. This is evident in the India where there are at least nine different extant Brahmic scripts (Salomon 1998) used by the major languages, belonging to Indo-European, Dravidian and other families (Wagner et al. 1999, p. 24). Scripts for languages such as Tibetan (Tibeto-Burman, Tibet), Burmese (Tibeto-Burman, Myanmar), Sinhala (Indo-European, Sri Lanka), Balinese and Javanese (Austronesian, Indonesia), Thai (Thailand, Austro-Thai), Khmer and Lao (Laos, both Mon-Khmer) belong to the same script class and have similar issues. There is no easy and widely acceptable text-entry method for most of these languages. This is true even in the case of Hindi, which is used natively by between 182 and 366 million people, and Bengali used by between 181 and 207 million people. Thus, one of the problems that makes the use of computers by a large segment of the world's population very difficult, perpetuating a severe but unacknowledged form of digital divide, is the lack of adequate text-entry methods. Currently, text-entry in these languages is done only by professional typists with months of training, or by dedicated individuals.

The rest of the paper discusses the nature of Brahmic scripts, and follows it with a brief description of existing text input schemes for Unicode fonts, our approach to implementation of a text-input scheme for an exemplar language, our evaluation approach, and future work we intend to pursue.

2 Nature of Indic (Brahmic) Scripts

According to (Coulmas 1990), the Brahmi script and Brahmi-derived (Brahmic) scripts have the following characteristics:

1. The scripts have signs for word-initial vowels.
2. Every basic sign has a consonant and the vowel */a/* (like *schwa* in English) as value.
3. Other vowels are represented by modifying the respective consonant with a diacritic mark.
4. Consonant clusters are represented by ligatures, all but the last consonant lose their inherent vowel.
5. The inherent vowel /a/ can be muted by a special diacritic.

There are two main classes of Brahmic scripts, Northern and Southern, each group encompassing several scripts. All these scripts are built on the

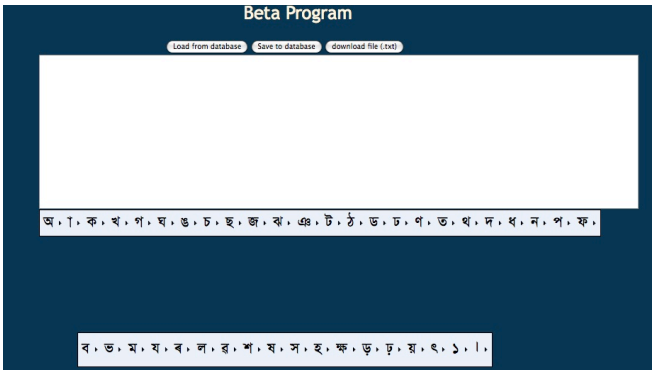


Figure 1 *shows the top-level menu items for character entry. The 11 vowels are represented by the first menu item in the top row. The second item in the top row represents the medial diacritic representations of the eleven vowels. The last three items on the bottom row represent special characters, numerals and punctuations, respectively.*

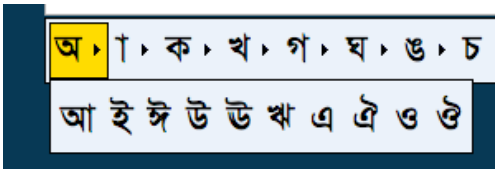
same principles and some are almost isomorphic, but they are not so similar that having learned one enables one to read others.

The Devanagari script is the most widely used script in India. Another important Brahmic script in Assamese-Bengali used by Bengali (about 200 million speakers), Assamese (30 million speakers), Meitei (1.5 million speakers) and Bishnupriya Manipuri (0.5 million speakers). Of the languages using Eastern Nagari script, 3 are among the 22 official languages of India. In this paper, we use Eastern Nagari script as an exemplar script.

In all Brahmic scripts, consonant clusters are represented by ligatures. The ligatures have to be learned separately because in many cases their sound value is hard to infer on the basis of the graphical composition of the complex letter sign (Columas 1990). Medial vowels (i.e., non-initial vowels or vowels positioned between letters) are written in the form of diacritic marks, with certain inconsistencies regarding positioning.

In the Eastern Nagari script, there are 11 word-initial vowels and diphthongs, 11 word-medial vowel diacritics, 37 semi-vowels and consonants, at least 240 ligatures, four special characters, 10 digits, and several special punctuation marks. The medial vowel forms appear inconsistently, sometimes above, sometimes below, sometimes to the left, sometimes to the right, and sometimes on both sides of a consonant or a ligature. Even for a single vowel, there may be more than one form of diacritic. The number of ligatures used varies from font to font. The additional complication is that the medial vowel forms and a few semi-vowel forms can potentially be used with any of the 37 consonants or semi-vowels as well as the 240 ligatures. There are at least twelve diacritic marks that can occur with at least 277 character forms. Thus, potentially, there can be several thousand characters or character-combinations that may appear. We have counted a total of at least 10 + 10 + (37 + 240) * 12 + 10 + 3 =3,357 character forms that can appear. In addition, the standard QWERTY keyboard based punctuation symbols can occur. There is also propensity to write English words in the middle of a vernacular text. Frequently, the numeric symbols are written in Roman. Thus, our challenge is to develop a text-entry system that can compactly represent the possibility of entering about 3000 characters as well as allow for Roman input.

Figure 2: Vowel menu showing 11 word-initial vowels.



As a first step, we have decided to represent the keyboard fully in software form. Not only should the software representation be efficient space-wise, the organization of the soft keys should look natural to a native speaker, entry of text using the softkeys should be efficient. Finally, dedicated users should be able to gain fast typing speed with practice.

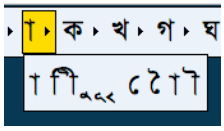


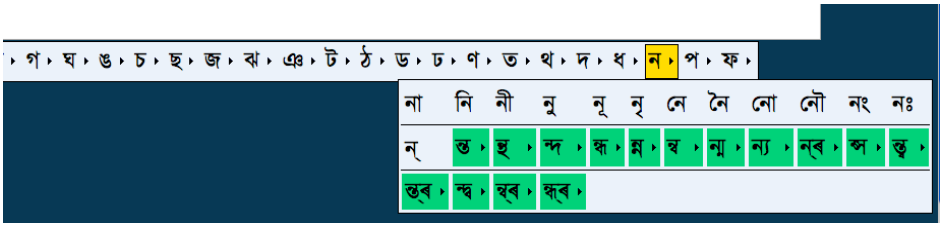
Figure 3: Vowel diacritics

3 Existing Text-Input Schemes

Text entry in these languages is almost impossible for most speakers, even ones who regularly communicate in these languages informally and informally. Only professional typists or dedicated individuals enter text in these languages. The commonly available entry schemes include

1. Dedicated keyboards: These allow text entry in a specific script, need special training to use, are almost impossible to find in the marketplace, and do not allow text entry in Roman character set. See (Joshi et al. 2004) for a discussion of a Devanagari keyboard.
2. Adaptation of QWERTY keyboards: These provide mapping of the QWERTY keyboard to characters in a language. Usually the mapping uses shift, option, command and other keys to obtain the variety and number of keys needed. The mappings are inconsistent across font makers. There is no correlation between the charac-

Figure 4: Second level menu (light blue in green) for the consonant ন (na).



ters represented by the direct keys and the ones with modification. Leisher (1996) discusses text input methods for ten languages including Thai by adapting the QWERTY keyboard.

3. Softkeys: Different Web sites have created softkeys for entering text. The softkeys list the basic vowels, consonants and semi-vowels and special characters.

In none of the above approaches, there is an easy way to enter the ligatures that are abundantly used in these languages, making text entry slow and painful. In the dedicated keyboard scheme, the typing of ligatures is tedious. When softkeys are provided on a Web site, usually only the basic characters are provided for clicking and entering into a textbox. Out of the 250 or so ligatures commonly used, none is usually provided for direct entry. To the type a complex ligature, which is written by most people as one single glyph if they are writing by hand, has to be written using up to five key presses. The hasanta is a way to tell the font processing system under the Unicode scheme to say that we are in the middle of creating a multi-consonant ligature where a consonant will be followed by another.

4 Pop-up Menu-based Text Entry Scheme

We have decided to extend the commonly available softkey schemes so that a user can easily and efficiently enter the large number of glyphs, the special cases for glyphs and vowel diacritization, as well as it make it easy for entering vowel dicritization with consonants, semi-vowels and ligatures.

Figure 1 shows the basic layout. Figure 2 shows the word-initial vowels from a pop-up menu. Figure 3 shows the diacritic marks corresponding to the vowels.

Figure 4 shows a second level pop-up menu corresponding to the character ন (na). There are three rows in the pop-up. The first row is mostly the vowel diacritics except the last two that are special diacritics. The first character of the second row is the hasanta dicritic that allows one to build ligatures one by one. The rest of the characters in the second and third columns represent two or three-consontant ligatures that usually take several

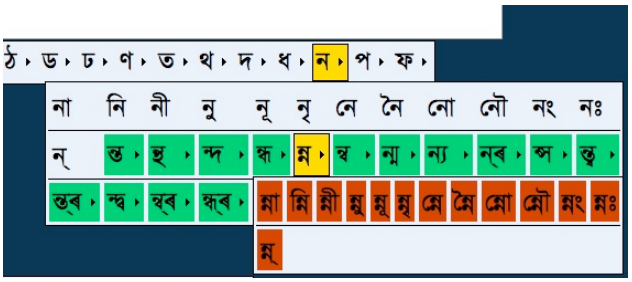


Figure 5: Third level menu (in red) for a ligature.

Unicode characters to type. Figure 5 represents an example third-level menu where we provide for vowel diacritization of the ligatures in addition to a few special diacritization.

Table 1 provides a summary of the numbers of pop-up menus at various levels. The number of pop-ups varies from two to 43 at the second level. The mean number of pop-ups at level 2 is 18. Out of 34, 42 characters, i.e., 81% have second level pop-ups. The third level pop-ups are for the vowel diacritic marks and a few other diacritics.

Table 1: No on pop-up menus at various levels

Item	Number of pop-up menus
Top-level	42
Mean no pop-ups at level 1	18
Max no pop-ups at level 1	43
2nd highest no of pop-ups, level 1	29
3rd highest no of pop-ups, level 1	28
4th highest no of pop-ups, level 1	24
No of chars with 2nd level pop-ups	34
Mean no of pop-ups at level 2	7
Max no of pop-ups at level 2	16
2nd highest no of pop-ups, level 2	11
3rd highest no of pop-ups, level 2	10

5 Implementation

We implemented the on-screen-keyboard using common Web technologies like PHP, JavaScript, HTML and MySQL. We used the Sothink DHTML library to dynamically calculate the *x* and *y* position for every component. We decided to place two menu bars, the upper menu contains 24 characters and the lower menu contains 18 charac-

ters. They are separated by several pixels, which allows for an empty "space" to display the on-hover effect. In order to record metrics, we used Ajax to communicate with the backend server and databases. Scripts were developed to track the user input. The motivation behind this implementation is that we can create on-screen-keyboards for different languages by only populating the databases.

6 Evaluation

We decided to utilize common metrics for measuring text entry performance (Wobbrock 2004, MacKenzie 2004): characters per keystroke, average number of keystrokes/minute, average number of characters/minute, and the amount of movement of mouse or trackpad.

We tested our text input scheme with 5 testers distributed around the US and India. The text was chosen randomly from two encyclopedias published after 2000. During a period of one week, our testers typed 10,748 characters. Table 2 shows that 50% of clicks produce 2+ Unicode characters.

Characters per minute is a common metric for measuring text input. Our testers are able to type 30 characters per minute on average. We did not ask our testers to type in a single sitting or not to multi-task. It translates to about 7-10 words per minute. This is consistent with the speed attained by subjects for various softkey layouts for English (Mackenzie et al. 1999). We conclude that the results so far show that on-screen-keyboard enables text entry at a reasonable speed by an average computer user for a language for which text entry has been very difficult. We also believe that it has the potential to allow for much faster text entry input with additional work.

Current results show that the averages distances (in pixels) that the user travels between keystrokes are: 270 along X-axis, 95 along Y-axis and 302 along XY.

7 Summary and Future Work

We want to experiment by changing our interface so that the average distance between clicks is reduced. We also want to study the possibility of allowing text input combining some QWERTY keyboard input and on-screen-keyboard technology. We also want to implement word prediction algorithms to speed up the process of text input. We have access to a corpus of about half a million words for implementing word prediction algorithms (Sharma et al. 2008). The corpuses are in Romanized form; we are in the process of converting them to Unicode. Finally, we would like to use the lessons learned to produce text entry interfaces for other Brahmic scripts.

Table 2: No of Unicode characters per keystroke

No of chars/click	No of characters	Percentage
1	5372	50.0
2	4184	38.9
3	663	6.2
4	469	4.4
5	40	.4
6	20	.2

References

Columas, F. 1990. The Writing Systems of the World, Basil Blackwell, Cambridge, MA.

Joshi, A., Ganu A., Chand A., Parmar V., and Mathur G. 2004. Keylekh: A Keyboard for Text Entry in Indic Scripts, *CHI*, Vienna, Austria.

Leisher, M. 1996. Input Method Design, *9th International Unicode Conference*.

MacKenzie, I.S. Evaluation of Text Entry Techniques, in *Text Entry Systems: Mobility, Accessibility, Universality*, Morgan Kaufmann, pp. 75-101.

Mackenzie, J.S., Zhang, S.X., and Soukoreff, R.W. Text Entry Using Soft Keyboards, Behavior & Info Technology, 18:4:235-244.

Salomon, R. 1998. Indian epigraphy: a guide to the study of inscriptions in Sanskrit, Prakrit, and the other Indo-Aryan languages.

Sharma, U., J. Kalita and R. Das. 2008. "Acquisition of Morphology of an Indic Language from Text Corpus," *ACM TALIP*, 2008, pp.1-33.

Wagner, D., Venezky, R. and Street, B. 1999. Literacy: An international handbook.

Wobbrock, J.O. 2004. Measures of Text Entry Performance, *Text Entry Systems: Mobility, Accessibility, Universality*, pp. 47-74.