Hindawi
Mobile Information Systems
Volume 2020, Article ID 3461651, 14 pages
https://doi.org/10.1155/2020/3461651

Research Article

Analysis and Evaluation of Braille to Text Conversion Methods

Sana Shokat,[1] Rabia Riaz,[1] Sanam Shahla Rizvi,[2] Khalil Khan,[1] Farina Riaz,[3] and Se Jin Kwon[4]

[1]*The University of Azad Jammu and Kashmir, Muzaffarabad 13100, Pakistan*
[2]*Raptor Interactive (Pty) Ltd., Eco Boulevard, Witch Hazel Ave, Centurion 0157, South Africa*
[3]*University of Southern Queensland, Toowoomba, Australia*
[4]*Department of Computer Engineering, Kangwon National University, Samcheok 25806, Republic of Korea*

Correspondence should be addressed to Se Jin Kwon; sjkwon@kangwon.ac.kr

Received 14 February 2020; Revised 5 May 2020; Accepted 4 June 2020; Published 26 July 2020

Academic Editor: Ali Kashif Bashir

Technology is advancing rapidly in present times. To serve as a useful and connected part of the community, everyone is required to learn and update themselves on innovations. Visually impaired people fall behind in this regard because of their inherent limitations. To involve these people as active participants within communities, technology must be modified for their facilitation. This paper provides a comprehensive survey of various user input schemes designed for the visually impaired for Braille to natural language conversion. These techniques are analyzed in detail with a focus on their accessibility and usability. Currently, considerable effort has been made to design a touch-screen input mechanism for visually impaired people, such as Braille Touch, Braille Enter, and Edge Braille. All of these schemes use location-specific input and challenge visually impaired persons to locate specified places on the touch screen. Most of the schemes require special actions to switch between upper and lowercase and between numbers and special characters, which affects system usability. The key features used for accessing the performance of these techniques are efficiency, accuracy, and usability issues found in the applications. In the end, a comparison of all these techniques is performed. Outcomes of this analysis show that there is a strong need for application that put the least burden on the visually impaired users. Based on this survey, a guideline has been designed for future research in this area.

1. Introduction

Visually impaired people are an important part of every community [1]. They are also concerned in learning the details of everything they encounter in their daily life [2]. The total number of visually impaired people is 2.2 billion; among them, 36 million are completely blind, and rest of the 1 billion have moderate to severe vision impairment [3].

Approximately thirty-seven million of the six billion populations worldwide are suffering from blindness. Unfortunately, 80% of blind people live in developing countries with restricted facilities for them [4].

Smartphones have become an integral part of everyday life. An expected increase of smartphone users will increase up to nine billion by 2022 [5]. The widespread use of smartphones has brought significant changes in how people learn. Research indicates that about one-third of smartphone usage consists of educational activities. Although smartphone usage has increased exponentially, it has low prevalence among people with visual disabilities. There are many complex accessibility issues that must be resolved in order to enable the full inclusion of this community [6]. Accessibility issues have been an important research domain over the last few years promoting the development of thousands of smartphone applications to help people with a visual disability, e.g., voiceOver services, talkback services, screen readers, and navigators.

These researches resulted in a dramatic increase in mobile-screen reader usage for the visually impaired, from 12% in 2009 to 88% in 2019 [7]. Despite the benefits that smart devices can offer, if the learning applications are not properly designed, their touch-screen interfaces may place an extra burden on blind learners. There are features such as VoiceOver for iPhone that help blind users interact with their device and browse content. However, educational applications often fail to consider the interaction patterns of blind learners with smart devices.

The language visually impaired people use for reading and writing is known as Braille, which was designed by Louis Braille. It is composed of six raised dots that can be easily written by visually impaired [8]. His design is illustrated in Figure 1. Each Braille character is represented using a combination of six dots arranged in a 3 by 2 matrix [9].

The Braille code system has been widely adopted in several communities because of its simplicity and comfort. Braille has been supported by different languages such as English, Arabic, and Hindi, among others [10]. However, few studies have been conducted on Braille for smartphones.

Research on Braille to text conversion has been carried out in the USA, Canada, India, Pakistan, and France. The literature shows that majority of the conducted research is limited to the USA and Canada. This indicates that there is a considerable demand for such a study in the rest of the world [6].

With the advancement of technology, Braille scripting mechanisms became an important research domain. Within this category, an initial device called Perkins Brailler was introduced to facilitate Braille writing. Space, backspace, and line space keys were designed in Perkins Brailler, as well as keys corresponding with each of the six dots in the Braille code [11]. In 2008, a lighter and quieter version was developed and launched that included an erase key and integrated carrying handle, which was not available in Perkins Brailler.

Another adaptation of the Perkins Brailler, the SMART Brailler, was created by David S. Morgan and released in 2011 [12]. Along with the existing features, Smart Brailler also included text-to-speech functionality in several languages. With the advent of computers, many users created Braille output by connecting a computer and Braille embosser. Visually impaired users were able to read the computer screen using screen reader computer software and/or Braille displays. Another similar Braille recognition system was designed by [13]. In this scheme, images were distributed into three threshold values, and Braille characters were subsequently recognized. Effectively, this interpretation was used to create a suitable dictionary. Recent research has focused on eliminating the need for separate hardware in Braille scripting. Application-level software has been designed to facilitate Braille users.

This survey focuses on gathering the difficulties faced by visually impaired while using a computing-based Braille input mechanism. Many technology-oriented applications for the visually impaired are available. Only those applications are considered for these surveys that are part of research taking place in different countries. Studied applications were analyzed and compared based on matrices related to usability issues for touch-screen-based Braille input methods. These evaluations bring forward the strengths and weaknesses of current schemes with a special focus on usability. No such study exists in the literature, so this paper can provide guidelines to the researchers for designing future applications that are highly usable for visually impaired people.

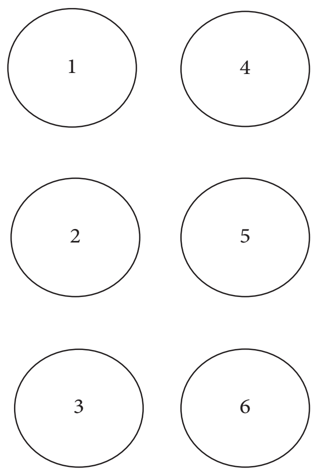

Figure 1: Braille Dots.

This paper comprises the following sections. Section 2 gives a detail insight into the previous studies that have been designed for entering Braille data. Section 3 describes the methodology in detail. Input methods are compared and evaluated in Section 4. Usability issues gathered from different studies are also included in this section. Section 5 concludes the paper and gives future recommendations.

2. Previous Work

This survey paper provides a review of the current state-of-the-art Braille input methods. In this section, we provide a detailed insight into the problems with these schemes when these are used by visually impaired people. Based on the current survey, we have identified new directions for future research. In recent years, many studies have been conducted to make Braille more technology-assisted. To analyze these studies, we have broadly divided the Braille input mechanism into two main categories:

(i) *Scanned Input*

(ii) *Touch-Screen-based Input*

2.1. Scanned Input. In the scanned Braille input, Braille Dots are extracted from Braille sheets using a scanner and, then, converted into text using optical character recognition, as shown in Figure 2. In this mechanism, visually impaired users give input on sheets without any interaction with a computing device.

2.1.1. Arabic Optical Braille Recognition System. A study was conducted that takes input from a flatbed scanner, as it clearly displayed the Braille Dots. The scanned image was converted into grayscale, the image frame was cropped, and the resulting image was stored in a 2D array. To remove the skewness in the framed image, an algorithm was designed. Finally, the Braille cells were recognized. They achieved approximately 99% accuracy for Braille written in Arabic with single- and double-sided scanned documents. They did not evaluate their system against any other application [14].

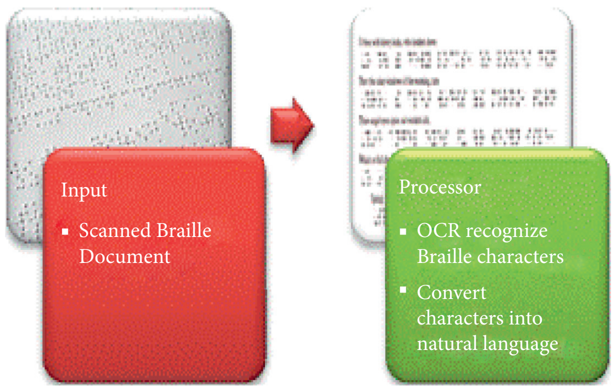
Input
Scanned Braille Document
Processor
OCR recognize Braille characters
Convert characters into natural language

Figure 2: Scanned Braille input.

2.1.2. Deterministic Turing Machine for Context-Sensitive Translation. To reduce the written communication gap between visually impaired and sighted people, a mechanism was developed to convert Braille codes into Urdu text [15]. They used a scanned Braille document as the input, which was, then, converted into a binary grayscale image. Multiple experiments were conducted for adjusting the threshold value of the scanned document. The primary focus of this study was to evaluate the context- sensitivity of Urdu-Braille. The main issue in the development of Urdu-Braille is that Braille is written left-to-right, whereas Urdu is written right-to-left. This issue was resolved by inventing a deterministic Turing machine that translates Urdu into Unicode and, then, reads the Unicode from left-to-right.

2.1.3. Text Translation of a Scanned Hindi Document into Braille via Image Processing. Hindi to Braille conversion was performed so that visually impaired people could have access to a wider range of Hindi literature [16]. They designed a database for consonant and matras generation using an image segmentation technique. Segmented letters were matched with the generated Hindi database using Principal Component Analysis before conversion into equivalent Braille codes. Braille cells for the Hindi text were generated with the help of mapping tables. Every image in the database was sized to 187×128 pixels to minimize the memory requirement. This scheme also requires less programming time as it utilizes a letter position in the Hindi database for letter matching.

2.1.4. Braille Instant Translator. The Braille Instant Translator converts Braille documents into the English language by identifying Braille Dots embossed on a page [17]. The scanned Braille document uses a camera-generated image as input and identifies the Braille Dots to emboss on the page in a grayscale image format. The extracted image was enhanced through standard deviation evaluation. A histogram was generated from the pixel values for image binarization. The brighter regions were assigned a value of 1, and the remaining regions a value of 0. MATLAB image-processing techniques were used to identify dots that were, then, matched with the English alphabet. Effectually, results indicated that the study achieved 80% accuracy, but implications are limited to only Grade 1 Braille.

2.1.5. Web-Based Urdu-Braille Translator. A web-based Urdu-Braille translator was developed by Iqbal et al. [18] for parents of the visually impaired. The translated Urdu script can be exported to PDF or directly sent to an embosser. Four modules were developed to enable an explicit learning process: a multiplechoice interface, an Urdu-Braille Reading module, a Braille-to-Urdu word mapping module, and a fill-in-the-blank module for Braille-to-Urdu word translation. To prove the effectiveness of their Urdu-Braille translator, a usability study was conducted on the parents of 15 blind people. For these tests, the participants used the Urdu script from BBC Urdu and translated it with the Urdu-Braille translator. Results indicated that, with each new test, the number of correct answers increased, verifying the effectiveness of the translator when applied to Braille learning. Given the educational impacts of the translated Urdu script, parents were able to use the online translator's Braille-learning application for more involvement with their visually impaired kids.

2.1.6. Feature Extraction Using SDAE. Braille pattern feature extraction is a cumbersome task. To reduce complexity, Stacked Denoising Auto Encoder (SDAE) was applied for Braille recognition [19]. SDAE is a deep learning technique used for automatic feature extraction and dimension reduction. To create the initial dataset, the authors produced a sample dataset of Braille images by segmenting the original photos taken with a digital camera from a Braille book. This scheme used SDAE for Braille feature extraction in order to obtain initial weights. The output of SDAE was used as input in SoftMax to build the classifier for training, which employed a traditional supervised learning algorithm with an initial dataset. Consequently, a deep network was initialized, and then, weights were fine-tuned using a back-propagation algorithm. A comparison of three different networks was performed, including multilayer perceptron (MLP), radial basis function (RBF), and SoftMax. Results indicate that, in Braille recognition, SDAE outperforms the traditional feature extraction algorithms and SoftMax outperforms MLP and RBF in conjunction with SDAE. Results demonstrate that, when compared with traditional methods, deep learning techniques facilitate recognition of Braille patterns due to effective automatic feature extraction and a reduced preprocessing time.

2.1.7. Braille Translator: Braille to Speech Converter. An application was designed by Falcon et al. [20] that converts the scanned Braille document into text and, then, speaks the translated text. Dynamic thresh holding technique is applied in this scheme to extract the important information from the scanned document. Scan Braille Dots are then recognized using pattern recognition and recovered using the Braille grid. The final generated image is in the form of Braille Dots in proper lines. For character translation, each dot is read before being converted into the binary and speech format. The authors were able to achieve an efficiency of 99%. It took only 26 seconds to translate a two-sided document into text and speech.

2.1.8. ODIA Braille. Another application named ODIA (a language used in East India) Braille was designed by the authors [21]. In this scheme, a scanned Braille document was converted into text and vice versa using the image-processing technique. Initially, a greyscale image is acquired from the scanned document. This image is enhanced using feature extraction and Braille cell segmentation technique. After this, recognized patterns are stored in the database. This database is verified by using different Braille samples for conversion.

2.1.9. A Braille Recognition System by Using the Mobile Phone with an Embedded Camera. A mobile camera-based application is designed in [22]which captures a Braille image. Using this application, a visually impaired person can convert a Braille written text anywhere easily. Noise removal is performed to extract the important features from the image using segmentation and fast dynamic thresholding techniques. A grid, constituting the location of the dots, is converted into vector form, from which Braille characters are recognized and translated into English characters. In this scheme, the time required to convert an image depends upon the quality of the image. Images with more noise will take a longer time to process. This application guarantees a 100% noise reduction. After noise removal, it only takes 2 seconds for the conversion process.

2.1.10. Recognition of Ethiopic Braille Characters. A Mechanism to recognize Ethiopic Braille characters was designed in [23]. The authors designed a new skewness correction technique. In the first step, noise is removed, and then, segmentation is performed using the direction field to detect the exact regions of the Braille Dots. In the next step, skewness correction is performed. After this, important Braille cell values such as the height of the cell, the width of the cell distance between different cells, and Braille character lines are identified. Initially, half characters are detected and recognized. Then complete Braille cell formulation and translation is performed. They tested their prototype in MATLAB. An average accuracy of 96.5%–98.5% is achieved for poor to medium quality images, while 99.9% accuracy is achieved for good quality images. Currently, this scheme can only convert one-sided Braille documents.

Table 1 provides a comparison of these schemes for which the Braille input is extracted from scanned documents or camera images.

2.2. Touch Screen. In this method, Braille data is input using a touch screen. As outlined in Figure 3, these images are compared with the Braille dataset.

The matched Braille character is then conveyed to the user using various output techniques.

2.2.1. NavTap and Braille Tap. The NavTouch keyboard for mobile devices was developed by Guerreiro et al. [24]. This layout of the keyboard divides the alphabet into five rows, starting with a vowel each row. By performing navigation gestures in four different directions, users can navigate through these rows: up, down, left, and right. Both vertical and horizontal navigations are cyclical. Audio feedback is also provided in order to locate the desired letter. Three groups of five users, who had no previous experience in mobile text entry, evaluated designs. Users were first trained on the text entry method. Results indicated that NavTap outperformed other layouts. However, these keyboards require both hands for operation, which is difficult for visually impaired people.

2.2.2. No-Look Notes. Bonner designed No-Look Notes an eyes-free, gesture-based text entry system for multitouch devices. In this layout, characters are arranged on the screen in eight pie-shaped menus, in which each segment includes three to four alphabets that correspond to the international standard mapping a phone keypad to letters. A user can move his or her finger on the pie menu, and the voice feedback will respond to assist them in selecting the appropriate character. Gestures are used to enter spaces or undo any action. This layout was tested on a group of 10 visually impaired users and evaluated based on matrices, such as words-per-minute (wpm), as well as the relationship between speed, accuracy, and errors. Results indicated a 100% increase in wpm for No-Look Note as compared with VoiceOver, but both exhibited approximately the same error rate. Not only is this scheme specific to English text entry but also to users complaining about fatigue during the entry procedure [25].

2.2.3. V-Braille. V-Braille is a method that conveys Braille characters in mobile devices using vibrations [26]. In V-Braille, the screen is divided into six portions of 3 rows and 2 columns. Each area represents a single Braille Dot. The phone vibrates when the user touches a region that corresponds to the raised dots in the current character. For example, touching the area for dot 2 and 5 provides a strong vibration to help users identify screen dots that are vertically adjacent. V-Braille is a useful output method for deaf-blind mobile device users, since V-Braille only provides haptic feedback. To evaluate this design, a user study was conducted. There were 6 male and 3 female participants in the study. The first task assigned to the users was reading 10 random characters, and the second task was reading short sentences. The average time calculated for reading a single character was between 4.2 and 26.6 seconds. More than 50% of the users were able to read the characters in less than 10 seconds. The time calculated for reading a short sentence was 130–781 seconds. More than 70% of participants clearly understood the meaning of the sentence. Subsequently, a semistructured interview was also conducted to elaborate on the overall environment of the application. Participants reported feeling happy and relaxed using this application. Designers of Braille Play games [27] built upon V-Braille,

Table 1: Braille to natural language conversion using scanned images.

References	Tool learning complexity	Language supported	Efficiency	Technique used
[14]	Not applicable	Arabic	99%	Arabic OBR system
[15]	Not applicable	Urdu	Not applicable	Turing machine for context-sensitive translation of Urdu-Braille
[16]	Image segmentation technique	Hindi	Not applicable	Principal component analysis
[17]	Easy to learn, and the Output can be obtained in a pdf form or directly to the embosser	Urdu	Not applicable	Web-based Urdu-Braille translator
[18]	Easy to learn as only grade 1 Braille was used	English	80%	Image-processing techniques
[19]	Easy to learn	English	92%	SDAE using SoftMax
[20]	Easy to use. Text-to-speech facility is also available.	English	99%	Dynamic thresholding technique
[21]	Easy to learn	Odia	Successful one-to-one mapping from ODIA to Braille	Feature extraction and Braille pattern recognition
[22]	Easy to learn, and the mobile can be taken anywhere.	English	100% noise reduction	Mobile camera-based application
[23]	Not applicable	Ethiopic	98.5%	Direction filed tensors are used

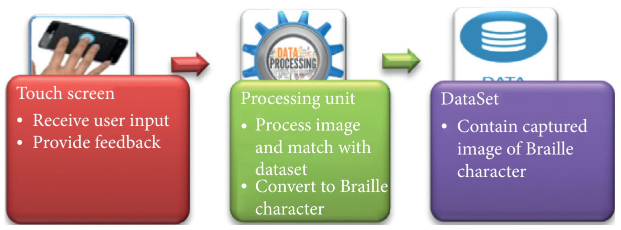
Touch screen
• Receive user input
• Provide feedback
Processing unit
• Process image and match with dataset
• Convert to Braille character
DataSet
• Contain captured image of Braille character
DATA PROCESSING

Figure 3: Braille input from a touch screen.

adding a speech output that tells the users which dot they are touching. They also created a V-Braille interface for entering characters, where the user double taps regions of the screen to raise the corresponding dots.

2.2.4. TypeIn Braille. TypeIn Braille, developed by Mascetti et al., tried to overcome the limitations of the existing keyboards. The TypeIn Braille keyboard uses gestures to enter Braille characters. Three different gestures of single- and multitouch were used for activating and deactivating Braille Dots. Audio or vibration feedback confirms the user input. Flick gestures were defined for input and editing options, such as the end of character, space insertion, text deletion, capitalization, text selection, and cursor relocation. Gesture memorization, time consumption for moving between writing, exploration, and selection mode are the basic drawbacks of the system [28].

2.2.5. Braille Touch. A study conducted on Braille keyboard design used six specific screen locations on touch-screen devices for entry of the Braille code [29]. Data entry required both of the user's hands, specifically three fingers from each hand for entry of a single character. Users faced problems simultaneously operating multiple fingers and both hands. To operate this keyboard, blind users must distance the phone from their faces, which can be uncomfortable and lead to a breach of privacy. Users were forced to write on the fixed locations mentioned on the screen, which burdens visually impaired people who struggle to keep track of the location. Several usability experts evaluated Braille Touch and two visually impaired users. The most common feedback was to provide a position-free text entry that did not require distance between users and their phones.

2.2.6. Braille Type. Oliveria et al. proposed a Braille Type. This keyboard was designed to avoid the multitouch technique [30]. Users must perform a single touch to enter a Braille Dot, and after a period, the user receives an audio confirmation. The timer for audio feedback can be adjusted per user's expertise. The user must double-tap the screen anywhere to accept the entered Braille character. The left swipe gesture is used to clear marked Braille cells. They evaluated Braille Type on 15 blind users. Results indicated a text entry speed of 1.45 wpm, a significant improvement from the 2.11 wpm achieved by VoiceOver. The error rate of Braille Type was comparatively low at 8.91% as opposed to VoiceOver's 14.12%. Overall results indicated that Braille Type, when compared with VoiceOver, was easy to learn and exhibited a low error rate, but performed slower. Due to less screen space, the integration of this keyboard with other applications is not possible.

2.2.7. Mobile Brailler. A new keyboard layout was designed based on the similar design patterns to those in TypeIn Braille [31]. Along with the tapping function, this keyboard also used swiping gestures for inserting and deleting characters. To enter two dots in a consecutive row, the user must tap with two fingers at the same time. To deactivate the left and right dots, a user must swipe to the right, which removes the last entered character. Mobile Brailler prototyped and compared five different input methods: One-Finger, Split-Tap, Two-Finger, Thumb-Typing, and Nine-Digit. Along with the tapping function, the right and left swipes are used for adding spaces and backspaces, respectively. The study was conducted on 15 visually impaired people from the greater New York City area, and the prototypes were tested on the android environment. Users answered qualitative and open-ended questions in each application. Results indicated that the One-Finger method was preferred by the visually impaired because of its simplicity. Gesture memorization was among the primary drawbacks. The typing speed of other techniques was faster than the single-finger method, but those techniques were not preferred due to gesture difficulty. Recommendations for these input methods were provided based on the user experience in order to improve prototype deficiencies.

2.2.8. Braille Key. Braille Key was designed with four large keys in the screen corners. Two buttons on the upper side are used to enter two columns of Braille code. A user must perform a single touch for entering the first dot, a double-tap to enter the second dot, and a long tap to enter the third dot. The lower two buttons are used for editing the text. The developed prototype was compared with Apple's well-established VoiceOver. Five visually impaired people participated in the study. Users were given a brief 10–15 minute awareness session on how to use both of the applications. Two sentences were entered without correcting any mistakes. Text entry speed and typing accuracy were measured for both applications. Effectively, Braille Key text entry speed and accuracy outperformed the iPhone's VoiceOver. The primary limitation of Braille Key is the identification of the button position on the screen [32].

2.2.9. Perkinput. Perkinput is a novel technique designed by Azenkot that uses the input finger detection (IFD) method. The input is entered into the device through multipoint touch. Tracking algorithms are used to detect the input finger reference point [33]. This method uses a 6-bit Braille code with audio feedback and provides single- and double-hand options. A study was conducted on eight users to evaluate this input method. Results found that Perkinput outperformed the iPhone's VoiceOver in speed and accuracy. A case study was conducted for performance evaluation, confirming that Perkinput improved writing proficiency without errors.

The Perkinput keyboard is particularly advantageous because it eliminates the fixed-key concept, successfully resolving navigational problems. However, creating a reference point requires users to single, double, and triple tap, the latter of which is time-consuming and often results in users forgetting the location of the reference point.

2.2.10. Braille Calculator. Learning mathematics is as important for the blind as it is for the sighted people. The Braille Calculator is designed so that visually impaired people can easily learn mathematics [34]. The input is taken from a touch-screen device. The Braille Calculator uses a 4-wire resistive 2.8-inch touch-screen. The screen is divided into six portions for text entry.

The touch-screen is interfaced with the Atmega 328 microcontrollers. An analog to digital converter (ADC) gives the user's impression of the location coordinates. The user is provided with an audio feedback in response to the dot entered. A step-by-step input is received for solving complex equations. Finally, the user is provided audio feedback regarding the solution. The Braille Calculator uses the Atmega 328 microcontrollers and a Secure Digital Card (SD) interface.

2.2.11. VB Ghost. VB Ghost, based on the Ghost word game, is an educational smartphone game for people with low or no vision [35]. The V-Braille interface was used as the basis of this game. The game was specially designed for educational and recreational purposes. This application was developed to reduce the accessibility issues that arise while using Braille on touch screens.

Both the hepatic and audio feedback were provided. Vibration occurs at the place where a raised dot occurs. In this game, a word fragment is presented and the player has to complete it. When satisfied with their letters, a player can submit the word by pressing the enter key or swiping with two fingers. The primary purpose of this game is to demonstrate the potential of developing fun, accessible, and educational games for visually impaired users.

2.2.12. Braille Play. VB Ghost was further improved and developed in the form of a complete suite named Braille Play. The suite consists of four different games: VBReader, VBWriter, VBHangman, and VBGhost. A longitudinal study conducted resulted in only one child capable of playing the game independently. However, some children were able to acquire the basic Braille-learning skills [27].

To analyze accessibility issues for blind people, four different smartphone applications were evaluated, namely, Blind Navigator, Easy Phone for the Blind, Blind Launcher, and Call Dialer [36]. To perform the study, ten visually impaired people were selected from various educational institutions. Ease of use, learning ability, absence of errors, efficiency, and voice understanding were the primary matrices for comparison. Survey forms were used for collecting feedback. Results indicated that, currently, most visually impaired people use Symbian phones, but blind people who already use smartphones are not ready to use any other device. Furthermore, 70% of the blind people easily understood the message in their Pakistani native language.

2.2.13. Eye Droid Keyboard. Another keyboard, Eye droid, was designed for entering Braille patterns using different gestures. To calculate the minimum swipe distance, input coordinates of the Braille Dots entered, that is, X1, Y1 and X2, Y2, are extracted, and the swipe threshold velocity is calculated using Velocity-X and Velocity-Y. The different gestures employed were left-to-right, right-to-left, and bottom-to-top swiping, as well as screen tapping [37]. These gestures, correspond with functions including activation and deactivation of left and right dots, activation of both dots, and activation of a single dot. The subsequent Eye droid-B scheme was compared with the earlier Eye droid-A design. Survey participants found Eye droid-B to be faster and easier to use when compared with Eye droid-A. This keyboard resolved navigational problems by eliminating the location-specific buttons that troubled users. Alternatively, users only need to memorize the defined gestures.

2.2.14. Edge Braille. Edge Braille introduces another text entry method that designed buttons in the touch-screen corners. This method allows a user to draw a continuous line by swiping along the screen edges for the entry of a specific character. When the user slides his or her finger on the screen, a vibrio tactile and voice feedback is returned to inform the user of Braille Dot activation or deactivation [38]. The input speed of Edge Braille was compared with TypeIn Braille and Perkinput. Results showed that the input speed of Edge Braille was faster than that of Braille Type and slower than that of Braille Touch. Furthermore, users found the Edge Braille interface easy to operate. However, numerous problems still exist with this design, since it is impossible to draw lines for all characters based on dot position. This method limits users to enter only alphabets and numbers and does not allow for "editing." Additionally, activation and deactivation of different dots reduces application speed.

2.2.15. Braille Easy. Braille Easy was developed for the entry of Arabic and English Braille codes within a mobile application. This system also used gestures, but for activation of the first and second column, users are required to tap once, twice, and three times [39]. This keyboard was significantly difficult for users to operate because it requires the memorization of different reference points. The keyboard speed was evaluated at 7 wpm, but the error rate was not specified. Furthermore, the keyboard only supports Grade 1 Braille.

2.2.16. Braille Ecran. A tactile interface cover for touch-screen phones was designed by [40]. This cover, called Braille Ecran, which provides an interface that consists of six Braille Dots. Each dot was given a tangible button that users can find and press. The significant advantages of this design include the "editing" capability, as well as vibration and audio feedback. However, users encountered considerable problems with the system, including confusion when operating buttons due to a single key's multifunctionality. Additionally, data entry was associated with a high probability of error due to the close proximity of buttons.

2.2.17. Single-Tap Braille. Single-Tap Braille is a position-free text entry method. A user can enter text, numbers, and punctuation by tapping anywhere on the screen. An algorithm runs in the background, interpreting the user's finger tapping for the identification of the specific corresponding character. The significant limitation of this model is the location-specific memory required; blinds often struggle to remember where they tapped once they have picked up their finger, eliminating their ability to complete subsequent actions correctly. No audio or vibrotactile feedback was provided in this scheme [41].

2.2.18. Braille Enter. Considering the problems highlighted in Single-Tap Braille, Braille Enter was designed to improve upon the method, consequently reducing on-screen navigation problems. In this method, the text is entered by activating and deactivating the Braille six-dot pattern, as shown in Figure 4. Additionally, the model allows for single-hand use and supports the entry of upper and lower-case letters, numbers, and special characters. Special functions are also available, such as adding and removing spaces. Furthermore, audio feedback was also provided to the users. The input received was used for activation and deactivation of the Braille Dots [42]. The press gesture entered the active dots, and a long tap deactivated the dots. The swipe function changed the character mode. The primary problem with Braille Enter is that users must enter all six dots even if only one dot is necessary, resulting in an excessively time-consuming process.

2.2.19. Braille Sketch. Braille Sketch, a gesture-based input method, was designed for visually impaired users on touch-screen devices [43]. A user simply draws a gesture for text entry. Audio feedback is provided when words are completed as opposed to letters to reduce time consumption. For error correction typing, an auto typing algorithm was used. A study was conducted on ten participants with visual impairments to evaluate the method. Each participant completed five typing sessions, and results demonstrated that Braille Sketch supports a text entry speed of 14.53 wpm with a 10.6% error rate.

In summary, in contrast to the immediate letter-level, Braille Sketch provides audio feedback to encourage users to type more quickly. To correct typing errors, an auto-correction algorithm is used.

3. Methodology

The research problem that motivated the conduction of this study was to highlight the usability issues faced by visually impaired people while using the latest technologies for Braille writing. This study also highlights a deeper understanding of which methods are in use for converting Braille into natural languages to enhance the scope of work performed in Braille language.

This is a survey-based research in which several searches were made to collect relevant research articles. These articles were collected from authentic resources such as Web of Science, IEEE Xplore, and Springer. Different queries were placed for searching such as the "Braille input method," "touch-screen-based Braille input method," and "touch-screen-based input method for visually impaired people." After skimming the results, only those papers were selected that take input from the visually impaired in Braille language, and then, they were processed into natural language. These research papers were

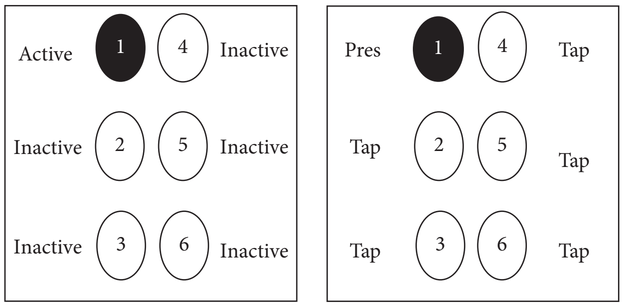

Figure 4: Braille Enter input screen.

broadly categorized as scanned-based and touch-screen-based on their input mechanism. In scanned-based, Braille sheets are scanned and input is given to the computer for further processing, while touch-screen-based directly took input from the visually impaired on touch-screen devices and, then, converted that input into its equivalent character.

After categorizing, a visit was made to the National Special Education Center (NSEC) "Mannak Payyan" which is the only local school for people with impairments. Both methods were discussed with the students of the National Special Education Center. For scanned input, since the Braille is written on paper, the students felt no difference in using computing technologies. Touch-screen-based input provided more motivation for the use of the latest technology. Thus, questions related to general user experience on entering Braille on touch screens were asked. A theoretical output was made after analyzing this raw data from which concern problems experienced by the visually impaired people were extracted and used as matrices for evaluating the usability issues. Therefore, we can design a system that helps the visually impaired community by fulfilling all their needs.

4. Comparative Analysis

A performance analysis was performed against VoiceOver based on the input efficiency of different techniques. On average, users require sixteen seconds to enter one word (approximately five Braille characters) using VoiceOver, which was designated as the standard [30].

In sixteen seconds, each method achieved the following word entries: Braille Type with 0.687 words, TypeIn with 1.211 words, Perkinput with 1.516 words, Braille Key with 0.524 words, and Edge Braille with 1.14 words. Evidently, TypeIn Braille and Perkinput outperform the other schemes with respect to data entry speed, as illustrated in Figure 5.

Table 2 summaries the efficiency and accuracy of Braille input schemes currently in use.

Our analysis found that Edge Braille exhibits not only the highest input efficiency at 7.17 wpm but also the lowest accuracy. Alternatively, V-Braille exhibits a low input efficiency (1.32 wpm) but achieves 90% accuracy. Braille Enter exhibits a similar pattern, achieving a text entry speed of 2.45 wpm with 85.88% accuracy.

These results indicate that, in order to achieve high accuracy, input efficiency is compromised. Research is

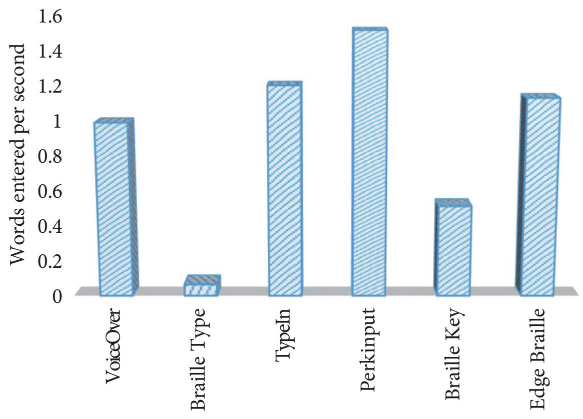

Figure 5: Braille input efficiency of different techniques.

needed to` design a Braille input technique that delivers satisfactory efficiency along with high accuracy.

We also analyzed how age affects learning Braille on touch screens. Most of the devices used for the assessment ran an android operating system. Figure 6 demonstrates that the age of the participant does not affect the learning time.

4.1. Usability Analysis. The usability analysis of touch-screen-based Braille input mechanisms is summarized in Table 3. This analysis was performed on different input schemes based on factors such as tool learning complexity, ease of use, feedback, language support, screen location dependency, and gestures used. Furthermore, two types of feedback, audio, and tactile were explored.

English is the most commonly used language in the current schemes. Visually impaired users found that applications that did not force them to touch specific locations for Braille Dot entry were easy to use. Alternatively, schemes requiring multitouch or memorization of a large number of gestures were found difficult to use.

All of these schemes were designed to enable visually impaired members of the society active participation in the advancing technology. These input methods focus on facilitating touch-screen Braille input for blind users. The primary disadvantages of the current schemes are discussed in the following sections.

4.1.1. Screen Location Identification. Keeping track of a specific location on a touch-screen device for entry of Braille Dots is a tiresome task for visually impaired users. Among these schemes, Edge Braille was considered the easiest to use, as the visually impaired were better able to identify the edges of the device.

4.1.2. Screen Location Identification. Keeping track of a specific location on a touch-screen device for entry of Braille Dots is a tiresome task for visually impaired users. Among these schemes, Edge Braille was considered the easiest to use, as the visually impaired were better able to identify the edges of the device.

Table 2: Performance analysis of touch-screen-based Braille input techniques.

Application name	Efficiency (wpm)	Accuracy/standard deviation	References
Single-Tap Braille	4.71	11.23	[6]
Braille Tap	3.35	3 MSD	[24]
Braille Key	1.8	5 MSD	[23]
Multitap	0.78	15.6 MSD	[24]
NavTap	1.25	9.99 MSD	[24]
VBraille	1.32	90%	[26]
Braille Touch	6.3	—	[29]
Braille Type	1.45	46.15%	[30]
VoiceOver	2.11	29.40%	[30]
Mobile Brailler	2.1	—	[31]
Braille Key	1.8	5 MSD	[32]
Edge braille	7.17	15%	[38]
QWERTY	3.72	20.54	[39]
Braille Enter	2.45	85.88%	[42]

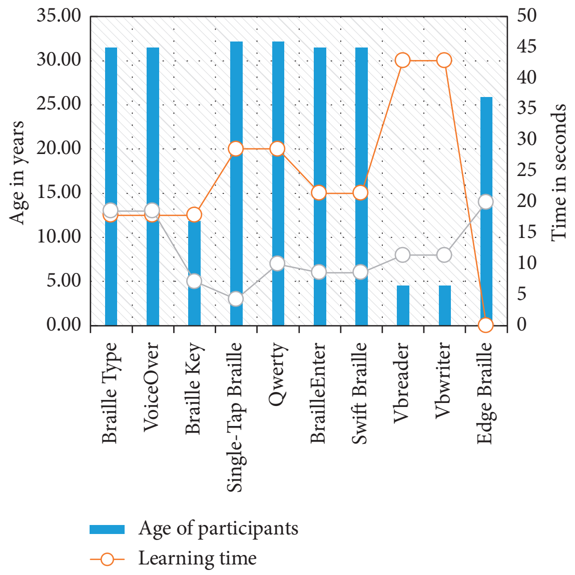

Figure 6: Relationship between the age of participants and learning time.

4.1.3. Use of Both Hands. Around 30% of the touch-screen-based schemes studied in this survey require that the visually impaired use both hands for entering Braille characters [24, 29, 31, 39]. This is a cumbersome task, especially if a blind user is walking or multitasking.

4.1.4. Multitaps. Long press, double-tap, and triple tap each require advanced usability of touch-screen devices. The following schemes require the use of multitaps for entering Braille text [28, 32, 38, 39, 42]. These mechanisms were found confusing to blind users, as mentioned by the examined literature.

4.1.5. Use of Gestures. The utilization of too many gestures significantly increases the Braille keyboard learning time, which was especially evident in [27, 28, 37, 42]. Specifically, mapping tasks with gestures, along with entering Braille Dots, caused a significantly higher error rate when tested on visually impaired users.

The provision of an eyes-free, comfortable text entry method is a vital research area. In current schemes, all the burden of correct Braille entry is placed on the visually impaired user. There is a substantial need for a mechanism that moves the burden from the visually impaired users to the technology, enabling greater accessibility and usability for this specific community.

On the basis of usability analysis, we have designed four new categories that can help researchers to design a better application for the visually impaired.

(i) Interactive display: for a visually impaired user, an interactive display would be one that is not screen-specific and also provides feedback. It could be an auditory feedback or tactile feedback

(ii) Efficiency of use: people with visual impairments find those applications more attractive that are less

Table 3: Comparison of touch-screen-based Braille input methods.

Sr #	Name	Sample size	Tool learning complexity	Usable	Single/double-handed usage	Tactile/audio feedback	Language supported	Allows editing	Screen location dependency	Gestures used	Technique used	References
1	Single-Tap Braille	7	Low	Low	Single	None	English	NA	No	Tapping	Algorithm for recognizing the taps	[6]
2	NavTap and Braille Tap	15	High	Low	Double	Audio	English	NA	Yes	(i) Up (ii) Down (iii) Left (iv) Right	Minimum string distance rate	[24]
3	No-Look Notes	10	N/A	Medium	Double	Audio	English	Yes	Yes	(i) Split-tapping (ii) Slide rule	*T*-test/Robust entry techniques	[25]
4	V-Braille	9	N/A	Medium	Single	Tactile	English	N/A	Yes	N/A	Semi-structured interviews	[26]
5	Braille Play	8	High	Low	Single	Audio	English	N/A	Yes	Moving between different options	Gesture recognition back-tracking algorithm	[27]
6	TypeIn Braille	7	High	Low	Both	Audio	English	N/A	N/A	(i) Single-Tap left (ii) Single-Tap right (iii) Double-Tap (iv) Triple-Tap	Usability study	[28]
7	Braille Touch	6	Low	Low	Double	N/A	English	N/A	Yes	N/A	Usability study	[29]
8	Braille Type	15	Low	Medium	Single finger	Audio	English	Yes	Yes	(i) Tap (ii) Left swipe	One-way ANOVA	[30]
9	Mobile Brailler	0	High	Medium	Both	N/A	English	Yes	Yes	Swiping	Euclidian distance for calculating touch points	[31]
10	Single-finger prototype	15	Low	High	Single	Audio	English	N/A	Yes	Tapping	Usability study	[31]
11	Braille Key	5	Low	High	Single	Audio	English	N/A	Yes	(i) Single-Tap (ii) Double-Tap (iii) Long press	One-way ANOVA	[32]
12	Braille Calculator	0	Low	Medium	Single	Audio	English	N/A	Yes	Tapping	Adriano UN board	[34]
13	VB Ghost	0	Low	Medium	Single	Tactile and audio	English	N/A	Yes	Tapping	Longitudinal study	[35]
14	Eye droid	12	Low	Medium	Single	Audio	English	No	No	(i) Swipe left (ii) Swipe right (iii) Swipe from bottom-to-top (iv) Swipe from top-to-bottom	Gesture recognition algorithm, experimental study.	[37]

Table 3: Continued.

Sr #	Name	Sample size	Tool learning complexity	Usable	Single/ double-handed usage	Tactile/ audio feedback	Language supported	Allows editing	Screen location dependency	Gestures used	Technique used	References
15	Edge Braille	14	Medium	Medium	Single	Tactile	English	N/A	Yes	Moving between different dots to draw continuous patterns	Lab-based study, ANOVA	[38]
16	Braille Easy	6	Medium	Low	Single	NA	English	N/A	NA	(i) Single-Tap (ii) Double-Tap (iii) Triple-Tap	N/A	[39]
17	Braille Enter	6	Low	Medium	Single	Audio	English	Yes	No	(i) Press (ii) Long tap (iii) Swipe	Braille enter algorithm	[40]
18	Braille Sketch	10	Low	High	Single	Voice	English	N/A	No	N/A	Auto correction algorithm	[42]
19	Braille Ecran	1	High	Medium	N/A	Tactile and audio	English	Yes	Yes	N/A	Predictive and experimental evaluation	[43]

Table 4: Usability-based categorization of current schemes.

Touch-screen-based input methods		New category
Single-Tap Braille	[6]	(1) Interactive display
Eye droid	[37]	
Braille Enter	[42]	
Braille Sketch	[43]	
Single-finger prototype	[31]	(2) Efficiency of use
Braille Key	[32]	
Braille Calculator	[34]	
VB Ghost	[35]	
Eye droid	[37]	
Edge Braille	[38]	
Braille Enter	[40]	
Braille Sketch	[42]	
Single-Tap Braille	[6]	(3) Easy memorization
Braille Touch	[29]	
Braille Type	[30]	
Mobile Brailler	[31]	
Single-finger prototype	[31]	
Braille Calculator	[34]	
VB Ghost	[35]	
Braille Sketch	[42]	
Braille Ecran	[40]	
No-Look Notes	[25]	(4) Recovery from errors
Braille Type	[30]	
Mobile Brailler	[31]	
Braille Enter	[40]	
Braille Ecran	[43]	

complex, can be handled using a single hand, and are simple to use.

(iii) Memorability: location-free specific applications or application that needs few gestures to remember are more appreciable by the visually impaired people.

(iv) Recovery from errors: applications that allow editing or reentering text helps in recovering from errors.

Table 4 presents the current schemes as per new categorization.

5. Conclusions

This survey paper focused on technological assistance available for visually impaired people. Braille input mechanisms can be categorically divided as scanned- and touch-screen-based input methods. In the scanned input, handwritten Braille sheets are scanned using scanners. Various studies have applied machine learning techniques such as optical Braille recognition, deterministic Turing machine for context-sensitive translation, feature extraction, and image processing to extract Braille Dots and convert them into a specific language. In the touch-screen-based input method, Braille Dots are entered using a touch screen on handheld devices such as mobile phones or tablets. Braille consists of six dots, and the basic mechanism of entering Braille using a touch screen requires entering Braille Dots by activating and deactivating pixels on the screen. Once the input is acquired, various algorithms process the extracted Braille Dots and convert them into their equivalent natural language characters or words. These touch-screen methods use haptic, audio, and tactile feedback to assist visually impaired users. This study compared different input methods on the basis of the entry speed and accuracies achieved, techniques used in the input methods, the number of participants on which the study has been conducted, gestures used, usability level, language used, feedback provided, and screen location independency. These comparisons enabled us to understand the strengths and weaknesses of the current applications. Based on this literature review, we plan to design an application that provides high usability to the visually impaired students. In the future, a newly designed application can be compared with the previous techniques to improve its performance. Machine learning techniques can be applied for acquiring better accuracy for Braille to text conversion.

Data Availability

Data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by Basic Science Research through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF2017R1D1A3B04031440).

References

[1] R. P. Mills, D. L. Budenz, P. P. Lee et al., "Categorizing the stage of glaucoma from pre-diagnosis to end-stage disease," *American Journal of Ophthalmology*, vol. 141, no. 1, pp. 24–30, 2006.

[2] P. M. Leonardi, "Innovation blindness: culture, frames, and cross-boundary problem construction in the development of new technology concepts," *Organization Science*, vol. 22, no. 2, pp. 347–369, 2011.

[3] WHO, *Blindness and Vision Impairment*, WHO, Geneva, Switzerland, 2018.

[4] Pakistan Today, 2.5 Percentage of Pakistanis Suffer from Blindness| Pakistan Today, Mindblaze Technologies, https://www.pakistantoday.com.pk/2011/10/14/2-5-percent-of-pakistanis-suffer-from-blindness/.

[5] Ericsson, "The ericsson mobility report," 2017, https://www.ericsson.com/en/mobility-report.

[6] M. Alnfiai and S. Sampalli, "An evaluation of SingleTap Braille keyboard: a text entry method that utilizes braille patterns on touchscreen devices," in *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, Galway, Ireland, October 2016.

[7] WebAIM, "WebAIMaccessibility in mind," 2019, https://webaim.org/projects/screenreadersurvey7/.

[8] L. Braille, *Method of Writing Words, Music, and Plain Songs by Means of Dots, for Use by the Blind and Arranged for them*, Institution Royale des JeunesAveugles, Paris, France, 1829.

[9] E. C. Jibril and M. Meshesha, “Recognition of Amharic braille documents,” in *Proceedings of the 5th International Conference on the Advancement of Science and Technology*, Chennai, India, 2017.

[10] D. Saad, Al-Shamma, and S. Fathi, “Arabic braille recognition and transcription into text and voice,” in *Proceedings of the 5th Cairo International Biomedical Engineering Conference*, pp. 227–231, IEEE, Cairo, Egypt, December 2010.

[11] J. O. Bickford and R. A. Falco, “Technology for early braille literacy: comparison of traditional braille instruction and instruction with an electronic notetaker,” *Journal of Visual Impairment & Blindness*, vol. 106, no. 10, pp. 679–693, 2012.

[12] L. M. Michelson, J. Kathleen, and D. Morgan, “Using a new electronic Brailler to improve Braille learning at the Florida school for the deaf and blind,” *Journal of Visual Impairment & Blindness*, vol. 109, no. 3, pp. 226–231, 2015.

[13] A. Antonacopoulos and D. Bridson, “A robust braille recognition system,” in *Document Analysis Systems VI*, Springer, Berlin, Germany, 2004.

[14] A. Al-Salma, Y. Al Ohali, M. Al Kanhal, and A. AlRajih, “An Arabic optical braille recognition system,” in *Proceddings of the 1st International Conference in Information and Communication Technology and Accessibility, ICTA*, Hammamet, Tunisia, January 2007.

[15] M. A. Fahiem, “A deterministic turing machine for context sensitive translation of Braille codes to Urdu text,” in *Combinatorial Image Analysis*, Springer, Berlin, Germany, 2008.

[16] U. Beg, K. Parvathi, and V. Jha, “Text translation of scanned Hindi document to braille via image processing,” *Indian Journal of Science and Technology*, vol. 10, p. 33, 2017.

[17] M. Z. Iqbal, S. Shahid, and M. Naseem, “Interactive Urdu braille learning system for parents of visually impaired students,” in *Proceedings of the 19th International ACM SIG ACCESS Conference on Computers and Accessibility*, Baltimore, MD, USA, 2017.

[18] S. Iqbal, A. Ali, M. Younus, M. Huzaifa, and Z. Abbas, *Braille Instant Translator*, pp. 327-328, National University of Computer and Emerging Sciences, Islamabad, Pakistan, 2017.

[19] L. Ting, X. Zeng, and S. Xu, “A deep learning method for Braille recognition,” in *Proceedings of the 6th International Conference on Computational Intelligence and Communication Networks*, November 2014.

[20] N. Falcon, C. M. Travieso, J. B. Alonso, and M. A. Ferrer, “Image processing techniques for braille writing recognition,” in *Computer Aided Systems Theory*, Springer, Berlin, Germany, 2005.

[21] K. Parvathi, B. M. Samal, and J. K. Das, “ODIA Braille: text transcription via image processing,” in *Proceedings of the 1st International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, February2015.

[22] S. Zhang and Y. Kazuyoshi, “A braille recognition system by the mobile phone with embedded camera,” in *Proceedings of the Second International Conference on Innovative Computing, Informatio and Control (ICICIC 2007)*, September 2007.

[23] M. Hassen and Y. Assabie, “Recognition of ethiopic braille characters,” in *Proceedings of the International Conference on Management of Emergent Digital Eco Systems*, ACM, Lyon, France, 2012.

[24] T. Guerreiro, P. Lagoa, P. Santana, D. Gonçalves, and J. Jorge, “NavTap and BrailleTap: non-visual texting interfaces,” in *Proceedings of the Rehabilitation Engineering and Assistive Technology Society of North America Conference*, Resna, Toronto, Canada, 2008.

[25] M. N. Bonner, J. T. Brudvik, G. D. Abowd, and W. K. Edwards, “No-look notes: accessible eyes-free multi-touch text entry,” in *Pervasive Computing*, Springer, Berlin, Germany, 2010.

[26] C. Jayant, C. Acuario, W. Johnson, J. Hollier, and R. E. Ladner, “V-braille: haptic braille perception using a touchscreen and vibration on mobile phones,” in *Proceedings of the 12th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2010*, Orlando, FL, USA, October 2010.

[27] L. R. Milne, C. L. Bennett, A. Shiri, and R. E. Ladner, “Braille Play: educational smartphone games for blind children,” in *Proceedings of the 16th International ACM SIG ACCESS Conference on Computers & Accessibility*, Rochester, NY, USA, 2014.

[28] S. Mascetti, C. Bernareggi, and M. Belotti, “TypeIn Braille: a Braille-based typing application for touchscreen devices,” in *Proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility-ASSETS'11*, Dundee, Scotland, October 2011.

[29] B. Frey, C. Southern, and M. Romero, “Braille touch: mobile texting for the visually impaired,” in *Universal Access in Human-Computer Interaction*, Springer, Berlin, Germany, 2011.

[30] J. Oliveira, T. Guerreiro, H. Nicolau, J. Jorge, and D. Gonçalves, “Braille Type: unleashing braille over touch screen mobile phones,” in *Universal Access in Human-Computer Interaction*, Springer, Berlin, Germany, 2011.

[31] N. Paisios, A. Rubinsteyn, and S. Lakshmi narayanan, “Mobile Brailler: making touch-screen typing accessible to visually impaired users,” in *Accessibility for Pervasive Computing*, Springer, Newcastle, UK, 2012.

[32] N. S. Subash, S. Nambiar, and V. Kumar, “Braille Key: an alternative Braille text input system: Comparative study of an innovative simplified text input system for the visually impaired,” in *Proceedings of the 2012 4th International Conference on Intelligent Human Computer Interaction (IHCI)*, IEEE, Kharagpur, India, December 2012.

[33] S. Azenkot, “Eyes-Free input on mobile devices,” ProQuest Dissertations and Theses, Washington, DC, USA, 2014.

[34] Y. V. Gidh, M. S. Latey, A. Roy, K. Shah, and S. Ingle, “Braille calculator,” *International Journal of Engineering and Computer Science*, vol. 2, no. 2, pp. 382–481, 2013.

[35] L. R. Milne, C. L. Bennett, and R. E. Ladner, “VBGhost: a braille-based educational smartphone game for children,” in *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility*, Bellevue, WA, USA, 2013.

[36] N. Sultan, K. Siddiq, T. Rashid, and M. Farooque, “Evaluation of smart phone applications accessibility for blind users,” *International Journal of Computer Applications*, vol. 127, no. 3, pp. 9–16, 2015.

[37] M. Shabnam and S. Govindarajan, “Braille-coded gesture patterns for touch- screens: a character input method for differently enabled persons using mobile devices,” in *Proceedings of the International Conference on Communication, Computing and Information Technology*, Chennai, India, 2014.

[38] E. Mattheiss, G. Regal, J. Schrammel et al., “EdgeBraille: braille-based text input for touch devices,” *Journal of Assistive Technologies*, vol. 9, no. 3, pp. 147–158, 2015.

[39] B. Sepic, A. Ghanem, and S. Vogel, *Braille Easy: One-Handed Braille Keyboard for Smartphones*, Qatar Computing Research Institute, Health Technology and Informatics, Qatar, UAE, 2015.

[40] J. Siqueira, F. A. Soares, and C. R. Silva, "Braille Ecran: a braille approach to text entry on smart phones," in *Proceedings of the IEEE 40th Annual Computer Software and Applications Conference*, IEEE, Atlanta, GA, USA, 2016.

[41] M. Alnfiai and S. Sampalli, "SingleTap Braille: developing a text entry method based on braille Patterns using a single tap," in *Proceedings of the 11th International Conference on Future Networks and Communications*, Quebec, Canada, 2016.

[42] M. Alnfiai and S. Sampalli, "BrailleEnter: a touch screen braille text entry method for the blind," *Procedia Computer Science*, vol. 109, pp. 257–264, 2017.

[43] M. Li, M. Fan, and K. N. Truong, "Braille Sketch: a gesture-based text input method for people with visual impairments," in *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, Baltimore, MD, USA, 2017.