I.J. Engineering and Manufacturing, **2016, 4, 32-39**
Published Online July 2016 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijem.2016.04.04

Available online at http://www.mecs-press.net/ijem

Braille Recognition using a Camera-enabled Smartphone

Gayatri Venugopal-Wairagade[a]

[a]*Symbiosis Institute of Computer Studies and Research, Symbiosis International University, Atur Centre, Gokhale Crossroad, Model Colony, Pune – 411016, India*

Abstract

The paper proposes a method to process the image of a Braille document that interprets the raised dots on the document and converts them to their equivalent English characters. It was found that under ideal conditions of light and alignment of the Braille document with respect to the smartphone, the application can achieve more than 80% accuracy. The application can be used in the education domain, wherein users who do not understand Braille may help visually-impaired or blind students in their learning activities such as assignments and tests.

Index Terms: Human computer interaction, braille, image processing, visually impaired.

1. Introduction

Braille is a system invented by Louis Braille which is used by visually impaired persons to read a document consisting of raised dots [1]. A braille sheet is a paper embossed with raised dots which are arranged in three by two cells as shown in Figure 1.

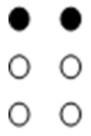


Fig.1. Image of a Braille cell wherein black dots are used to indicate the raised dots

The raised dots form a specific pattern, where each pattern corresponds to a particular character or a word and the reader identifies the pattern by touching the raised dots [2]. Braille is used widely by people belonging to various age groups. As a result of a survey carried out by Ryles, it was reported that the percentage of

* Corresponding author.
E-mail address:

employed visually impaired people who learned Braille as their primary medium was larger as compared to employed visually impaired persons who used print as their primary medium [3]. Braille is used even today by people with visual impairment, despite the availability of many digital alternatives [4]. According to a study conducted in 2012 [5], parents of visually impaired children believe that braille documents and books at home would help their child grow academically. It was found that the involvement of parents in children's studies was a significant factor in the children’s' academic achievements. Teachers who teach Braille to the students require training. A variety of teacher training programmes are conducted at various institutes to make the teachers proficient in Braille [6]. This paper suggests a solution that does not involve training, for teachers who are involved solely in the evaluation of answers written in Braille, and also for parents of visually impaired children, who are not well-versed with Braille. The application discussed, is intended to make the process of reading or learning braille easier for those who do not understand Braille notations. There are three grades in Braille [7], namely Grade 1, or un-contracted Braille, Grade 2 and Grade 3. In Grade 1 Braille, every character is identified by the pattern formed by the dot/s that is/are raised in the cell. Fig. 1 represents a Braille cell in Grade 1 Braille, where the two dots at the top are raised. Grade 2 and Grade 3 Braille use a pattern to represent a word, instead of a character. The authors proposes a smartphone application that would capture the image of a Braille document consisting of Grade 1 Braille, and convert it into its equivalent English text and would also generate the text to speech output based on the text displayed on the screen.

2. Related Work

A fair amount of work has been done in the area of optical Braille recognition using scanned images of Braille sheets. [8] proposed a method, wherein a Braille sheet is scanned and the scanned image undergoes a series of processes, such as image thresholding, edge detection, identification of dots, cropping cell frames, applying matching algorithm to the frames and finally producing the corresponding text and voice files. Canny edge detection is used, strong and weak edges are detected, and the edged dots are filled. Cells and words are determined based on the standard dimensions of a Braille sheet and binary codes are generated for each cell based on the presence or absence of a dot. The decimal equivalent was calculated. The decimal codes are mapped to the corresponding Arabic letter, and the equivalent text and voice files are produced. The paper suggests that future work on this should try to eliminate the need for manually setting the parameters for image processing. Here, the dot size and cell frame size were manually specified. [9] described a procedure where dots are detected using grids. According to the procedure, a grid is formed by generating horizontal lines from a particular point. Dots are detected after dividing the image into sub-images and then forming grids for each sub-image. The dots are subsequently grouped into a binary Braille character set, where each dot is represented by a bit position, and it is translated using a translation table. A similar method is described by [10], where an A3 scanner was used to scan double sided Braille documents. The image was divided into sub images, and grids were formed by plotting horizontal and vertical lines that go through the dots. The intersection of the lines was checked for the presence or absence of a dot. It was found that the time required to convert Braille to text was approximately 60 seconds for a single-sided sheet, and 80 seconds for a double-sided sheet. The error rate was low but it would increase or decrease depending on the quality of the sheet. [11] devised an algorithm to recognize the dots in Arabic Braille. Scanned images of single-sided braille documents were coloured and converted to gray-scale. Gaussian filter was applied to identify the noise in the image. The image undergoes thresholding to distinguish the dots from the background. The center of each dot is calculated and the dot is reduced to one pixel, which is located at the center of the original dot. This process takes 17 seconds to recognize sheets converted to green color and 21 seconds to recognize sheets converted to yellow color. [12] suggested a system where the image is thresholded such that it contains only three regions - dark, light and background. The dots are identified, rows of dots are identified, and a histogram is generated. The histogram is used to differentiate rows of dots within one cell from the rows of dots in another cell. The same process is carried out to differentiate a column of dots within a cell from column from a column of dots in another cell. A Braille dot grid is then constructed for each line. Braille patterns are recognized using the bit pattern, where each bit represents a dot position. These are then translated and printed. It has been argued by [13], that the scanned image has two distinct zones, one bright and one dark, above and below the dots respectively. Thresholding is applied on the image, based on the histogram peak. The dots are identified, based on the colour, the image is segmented. Parsers are used to translate text to Braille code, in this case, Portuguese Braille code. This method requires images of high quality, and provides different results when scanned using different scanners. Companies such as Neovision [14] and Adaptive Technology Solutions [15] have developed Optical Braille Recognition (OBR) Software, both of which are Windows software programs that take as input, the scanned image of a single or a double sided Braille sheet, and produce the equivalent text output on the screen. These methods use a scanner to capture the image of the Braille sheet. But if we want to render a Braille sheet in the absence of heavy equipment such as a scanner, we would require a device that is more compact and that can be used anytime irrespective of the location. Hence the use of a smart phone. But this would introduce more difficulties, as an image captured with the help of a smart phone camera may contain more noise and may contain non-uniformly illuminated regions, unlike that of a scanned image of the same sheet. [16] used a mobile phone with embedded camera, to recognize Braille and convert to equivalent Japanese text. After capturing the image with the mobile phone's camera, thresholding is applied on it. If more than two pixels surrounding the pixel being considered are not candidate pixels, i.e., they are not bright enough to be considered as being part of a dot, the single pixel being considered is removed. The areas, widths and heights of the dots are calculated, and the dots that are too small or too big are removed. If a dot is being considered and no other dot contains a similar radius, then that dot is removed. The dots are aligned horizontally, and a group is formed consisting of three lines (as it is a 3x2 cell). Based on the horizontal spaces between the dots, columns are created. The cell patterns are identified and mapped to the Japanese characters. The running time here, is directly proportional to the noise on the image. Also, the method analyzes the image pixel by pixel and makes comparisons, to identify raised dots. This could have a negative impact on the efficiency of the process, with respect to the time required to identify the raised dots. Thus there is a need for a system that is portable, compact, accurate and independent of the noise on the Braille document. A method is required to be devised, which is not only efficient, but also generates accurate output.

3. Approach

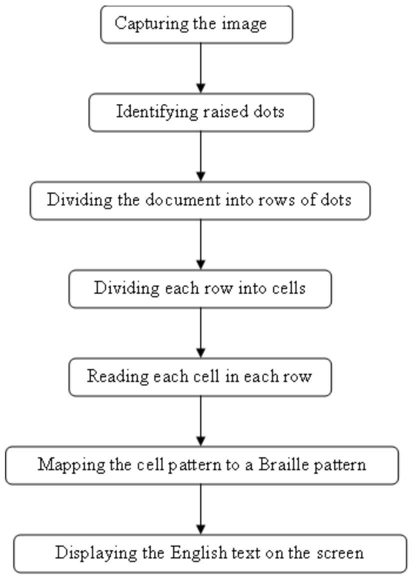

Capturing the image
Identifying raised dots
Dividing the document into rows of dots
Dividing each row into cells
Reading each cell in each row
Mapping the cell pattern to a Braille pattern
Displaying the English text on the screen

Fig.2. Outline of the process

The challenges with the objective of the study are associated with the use of the smartphone. Since the document is not scanned using a high quality scanner, the image obtained using the smartphone may be skewed, and could be of a low resolution, depending on the quality of the camera. This study focusses on the recognition of the raised dots on the document and their translation into equivalent English text. The entire process is divided into various sub-processes as shown in Figure 2.

3.1. Capturing the image

The image is captured using the camera on the smartphone. The image is taken in "sepia" mode, as it was observed that the dots are more prominent in "sepia" mode as compared to taking the image without applying any mode as shown in Figure 3.

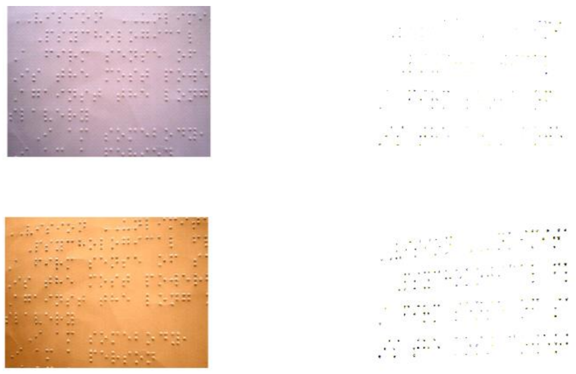


Fig.3. Difference between the images containing the recognized dots

3.2. Identifying raised dots

The next step is to distinguish the raised dots from the rest of the document as well as from any noise present on the document. This is done using the Open-CV library [17]. This library contains a variety of image processing APIs (Application Programming Interface). The circle detection API has been used here, which implements Hough's circle detection algorithm. After the dots are identified, all the dots are compared to each other and the maximum length and width of a dot are calculated. This is done in order to create a new image with dots of the same width and height.

3.3. Dividing the document into rows of dots

Once the new image is generated, in which all the dots have the same width and the same height, more processing is performed on this image, to identify rows of dots. This is done by reading the image from top to bottom and left to right. The position at which the first dot is found, is considered to be the top margin. We also recorded the position on the Y- axis, at which a dot is found as we progress along the X - axis. If this position is greater than the sum of previously recorded position and the width of the dot, this position is considered to be the bottom margin, and the area between the top margin and the bottom margin is considered to be a row. This process is repeated until the entire image is segmented into rows.

Algorithm:

Traverse the length of the sheet, starting from coordinates (0, 0) till current y coordinate equals the length of the sheet

```
For each coordinate on the Y axis,
traverse the breadth of the sheet, starting from the current coordinate on the Y axis.
If a dot is found
If no previous coordinate has been recorded,
Record the y coordinate of the current point.
Else
If the distance between the recorded coordinate and the y coordinate of the current point is greater than the sum of the recorded coordinate and the width of the dot,
Record the current y coordinate as bottom margin for that particular row.
```

3.4. Dividing each row into cells

Each row of the image is processed to identify the columns or cells of dots. It was observed that the horizontal distance between two dots in a cell is not greater than the width of a dot. This observation is used to divide the row into multiple columns. The horizontal distance between a dot and its subsequent dot is calculated. If this distance is greater than the width of a dot, the position at which the subsequent dot starts is marked as the end of the first column and the beginning of the second column. This process is repeated for all the rows identified in the previous step.

Algorithm:

```
Traverse the breadth of the sheet, starting from coordinates (0, 0) till current x coordinate equals the breadth of the sheet.
If a dot is found
If no previous coordinate has been recorded,
Record the x coordinates of the current point.
Else
If the distance between the recorded coordinate and the x coordinate of the current point is greater than the sum of the recorded coordinate and the width of the dot,
Record the current x coordinate as the end of the current column, and the beginning of the next column.
```

3.5. Reading each cell in a row

We have the top margin, bottom margin, left margin and right margin for each cell in a row. The top and bottom margins are used to divide a cell into three horizontal sections by identifying the positions of dots between the two margins. Similarly, the left and right margins are used to divide each horizontal section into two columns. Thus a three by two matrix of dots is obtained. Each cell is read, and a pattern of dots is obtained, using the positions of the dots within the cell. These patterns are stored in a data structure. For instance, if only one dot is found in the cell, and the dot is located in the top left section of the cell, then the pattern would be 100000, where 1 indicates presence of a dot, and 0 indicates absence of a dot. The number 1 at position 1 indicates that a dot is present at position 1 of a cell. Figure 4 depicts the numbering of positions within a cell.

1	4
2	5
3	6

Fig.4. Numbering of positions in a cell

After reading each cell, the horizontal distance between the current cell and the next cell is calculated. The relative distance between two cells is calculated to identify the end of the word, in order to add a space.

3.6. Mapping the cell pattern to a Braille pattern

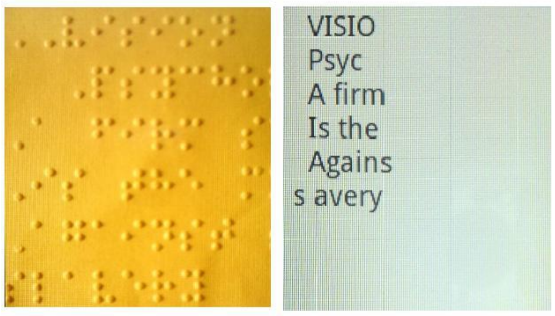
The pattern thus obtained is compared with the stored Braille patterns. Each pattern corresponds to an English letter, punctuation or a digit. If a match is found, the character that corresponds to the pattern is obtained. This process continues, and a string is formed.

3.7. Displaying the text on the screen

The string formed in the above step is displayed on the screen of the smart-phone. The text is read aloud using the default text-to-speech engine installed on the Android device.

4. Findings

The application was developed using the Android platform and it was found that the accuracy of the application varies greatly depending on the way the sheet is aligned with the smart-phone. Under ideal conditions of light and alignment of sheet with the smart-phone, an accuracy of over 80% was obtained. Figure 5 depicts the image and text obtained using the application. As can be seen, the image lacks clarity, that is, the dots are not prominently visible and appear slightly blurred.



```
VISIO
Psyc
A firm
Is the
Agains
s avery
```

Fig.5. Image of a Braille document and the text produced using the application

The words in the image are ‘VISION’, ‘Psycho’, ‘A firm s’, ‘Is the m’, ‘Against’, ‘slavery’. The image also contains case indicators that indicate the word or character is in upper case. The number of characters in the image is 44. Out of 44 patterns, 37 patterns were recognized correctly, giving an accuracy of approximately 84%.

5. Applications and further work

The application could be used in schools for the blind, to evaluate answer sheets, written in Grade 1 Braille. This would eliminate the cost incurred for training the staff in Braille. This would also open the doors of such schools to examiners who are not proficient in Braille. Parents and relatives of visually impaired persons could also be benefited by the application. This study is part of an exploratory experiment. The application can be modified such that it accommodates varying alignment and implements noise reduction algorithms. It can be enhanced to translate the Braille document into multiple languages, other than English alone. A similar approach could be used to interpret Grade 2 Braille.

Acknowledgements

The project was developed with the help and guidance provided by Mr. Kaushik Raghupathi, Mr. Nikhil Bhawkar and Mr. Shrenik Nikam. The author would like to thank Symbiosis Institute of Computer Studies and Research and Nowrosjee Wadia College, Pune for providing the necessary support and infrastructure for working on the project.

References

[1] Jiménez, J., Olea, J., Torres, J., Alonso, I., Harder, D., & Fischer, K. (2009). Biography of louis braille and invention of the braille alphabet. *Survey of ophthalmology*, *54*(1), 142-149.
[2] Pring, L. (1994). Touch and go: learning to read Braille. *Reading Research Quarterly*, 67-74.
[3] Ryles, R. (1996). The impact of braille reading skills on employment, income, education, and reading habits. *Journal of Visual Impairment and Blindness*, *90*, 219-226.
[4] Guerreiro, J., Gonçalves, D., Marques, D., Guerreiro, T., Nicolau, H., & Montague, K. (2013, October). The today and tomorrow of Braille learning. In *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility* (p. 71). ACM.
[5] Kamei-Hannan, C., & Sacks, S. Z. (2012). Parents' perspectives on braille literacy: Results from the ABC Braille Study. *Journal of Visual Impairment & Blindness*, *106*(4), 212-223.
[6] Scheithauer, M. C., & Tiger, J. H. (2012). A COMPUTER‐BASED PROGRAM TO TEACH BRAILLE READING TO SIGHTED INDIVIDUALS. *Journal of applied behavior analysis*, *45*(2), 315-327.
[7] AMERICA, B. A. O. N. (1994). English braille American edition. *Louisville, KY: American Printing Housefor the Blind*.
[8] Al-Shamma, S. D., & Fathi, S. (2010, December). Arabic Braille Recognition and transcription into text and voice. In *Biomedical Engineering Conference (CIBEC), 2010 5th Cairo International* (pp. 227-231). IEEE.
[9] Miesenberger, Klaus, Joachim Klaus, Wolfgang Zagler, and Arthur Karshmer. "Computers helping people with special needs." In *Proceedings of 12th International Conference on Computers Helping People with Special Needs (ICCHP 2010)*. 2010.
[10] Mennens, J., Van Tichelen, L., Francois, G., & Engelen, J. J. (1994). Optical recognition of Braille writing using standard equipment. *Rehabilitation Engineering, IEEE Transactions on*, *2*(4), 207-212.
[11] Authman, Z. I., & Jebr, Z. F. (2013). ALGORITHM PROTOTYPE FOR REGULAR FEATURE EXTRACTION OF ARABIC BRAILLE SCRIPTS. *International Journal*, *2*(1), 2305-1493.
[12] Antonacopoulos, Apostolos, and David Bridson. "A robust Braille recognition system." *Document Analysis Systems VI* (2004): 111-114.
[13] Shahbazkia, Hamid, Telmo Silva, and Rui Guerreiro. "Automatic braille code translation system." *Progress in Pattern Recognition, Image Analysis and Applications* (2005): 233-241.
[14] Neovision Homepage. [Online]. Available: http://www.neovision.cz.
[15] Adaptive technology solutions homepage. [Online]. Available: http://www.adaptivetech.co.nz.
[16] Zhang, Shanjun, and Kazuyoshi Yoshino. "A braille recognition system by the mobile phone with embedded camera." In *Innovative Computing, Information and Control, 2007. ICICIC'07. Second International Conference on*, pp. 223-223. IEEE, 2007.
[17] Bradski, Gary, and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'reilly, 2008.

Authors' Profiles

Gayatri Venugopal-Wairagade is an Assistant Professor at Symbiosis Institute of Comptuer Studies and Research, Symbiosis International University, Pune, India. She is involved in the development and administration of Moodle at the institute. She has taugh courses on Android and web technologies. Her areas of interest are accessible computing and education research.

How to cite this paper: Gayatri Abhishek Venugopal-Wairagade,"Braille Recognition using a Camera-enabled Smartphone", International Journal of Engineering and Manufacturing(IJEM), Vol.6, No.4, pp.32-39, 2016.DOI: 10.5815/ijem.2016.04.04