Journal of Computer Science 7 (11): 1749-1759, 2011
ISSN 1549-3636
© 2011 Science Publications

Dot Detection of Braille Images Using A Mixture of Beta Distributions

[1]Amany Al-Saleh, [2]Ali El-Zaart and [1]Abdul Malik Al-Salman
[1]Department of Computer Science, College of Computer and Information Sciences, King Saud University, Saudi Arabia
[2]Department of Computer Science, College of Computer and Information Sciences, Faculty of Sciences, Beirut Arab University, Lebanon

Abstract: Problem statement: Braille is a tactile format of written communication for people with low vision and blindness worldwide. Optical Braille Recognition (OBR) offers many benefits to Braille users and people who work with them. **Approach:** This study presents an algorithm for detecting dots composing braille characters in an image of embossed braille material obtained by an optical scanner. We assumed that a mixture of Beta distributions could model the histogram of a scanned braille document. The core of the proposed method was the use of stability of thresholding with Beta distribution to initiate the process of thresholds estimation. Segmented Braille image was used to form a grid that contains recto dots and another one that contains verso dots. **Results:** Braille dots composing characters on both single-sided and double-sided documents were automatically identified from those grids with excellent accuracy. **Conclusion:** The experiment showed that the proposed method obtained very good results but it requires more testing on different scanned Braille document images.

Key words: Optical braille images, dot detection, thresholding and beta distribution, Optical Braille Recognition (OBR), double-sided document Braille texts, segmented image, flatbed scanners

INTRODUCTION

Braille is a writing system that enables blind and partially sighted people to read and write through touch. It was invented by Louis Braille, a French teacher of the blind Information about Deaf Blindness. In its simplest form, Braille consists of letters, numbers and some punctuation marks. All kinds of material can be put into Braille, from bank statements to bus timetables, maps to music.

Historically, Braille materials were produced by hand using mechanical devices to press dots into heavy study. In the late 1960s, technology was developed allowing Braille texts to be entered, manipulated and stored using a computer (Braille Institute, 2011). At that time, computers still output text to devices that embossed metal plates; but later, study Braille embossers were developed that could be connected directly to a personal computer allowing much greater flexibility for the creation of Braille on a small scale.

Technology has shown great promise in providing access to textual information for people with low vision and blindness (Al-Salman *et al*., 2007). For some years, there has been an increasing trend to use computers for entering, editing and printing Braille documents using special purpose software and printers. An OBR allows reading volumes of typewritten documents with the help of flat bed scanners and OBR software.

OBR offers many benefits to Braille users and people who work with them such as facilitating communication, reducing storage space and preserving out-of-print Braille texts. Everyone who works with blind people and does not read Braille will benefit from using the OBR.

The components required for transferring this hard copy data to machine-readable material are the optical scanner hardware that performs the scanning and encoding of the data and the software that interprets these bits represented graphical images of the typed characters to their corresponding machine (ASCII code) representation. OBR works with single-side or inter-point (double-sided) Braille and can read both sides of an inter-point page with a single scan. In some cases, a camera is used instead of a scanner for the image acquisition of a Braille document.

There have been a number of efforts to identify cells Braille characters in Braille documents using different acquisition techniques. Relatively complex setups of a camera and lighting were used in (Hedgpeth

Corresponding Author: Amany Al-Saleh, Department of Computer Science, College of Computer and Information Sciences, King Saud University, Saudi Arabia

et al., 2003; Mennens *et al*., 1994; Mihara, *et al*., 2005). For example, for the system proposed in (Mihara *et al*., 2005), the size of the device and the existence of a camera over the user's head are found to be somewhat bulky and inflexible. In addition to the non-standard setup and equipment, the images obtained frequently suffer from the many problems of camera-based image acquisition (e.g., aberrations, irregular lightness, relatively low resolution) (Antonacopoulos and Bridson, 2004).

For this reason, we found that using a commercially available flatbed scanner may certainly provide a better and cost-effective solution. One of the earliest and comprehensive approaches to use a flatbed scanner is that of Bunke and Spitz (2006). Both single-sided and double-sided Braille documents are scanned at 100dpi with 16 gray levels produced. The proposed system performs few image-based operations and is relatively flexible to skew as it identifies Braille characters based on character-region search. A re-development of this approach was proposed by Antonacopoulos and Bridson (2004) where many improvements were added to increase the cost-effectiveness and usability of the system. Both approaches used a grid. However to handle the variation in positions of characters problem caused by the fixed grid Bunke and Spitz (2006), the authors in (Antonacopoulos and Bridson, 2004) used a flexible grid.

Mennens *et al*. (1994) addressed important issues related to using a scanner such as distortion and the shadows produced due to the tension in a Braille study which causes the study to be never flat. Solutions were provided. A gird is created for standard recto and verso dots and placed on the image where Braille dots are expected to be. However, it does not account for possible slight variations in character positioning in different lines; it assumes fixed locations of characters.

The new approach followed in this study offers several improvements on the previous approaches, while the cost effectiveness and usability of the system (using commercial scanner) remain high. Image preprocessing methods such as contrast enhancement, noise filtering, , are no longer needed. The core of the proposed method is the use of stability thresholding for a mixture of Beta distributions to initiate the process of a multi-mode estimator in order to create threshold values. A flexible grid is created to guarantee correct detection of Braille dots. The main steps involved in the development process are described as follows:

Image acquisition and understanding the braille image: an image of a single/double sided Braille page is obtained using a flatbed scanner. The Braille page is converted into a digitized image array. The pixel values of the digitized image are then modified and prepared for subsequent operations.

Segmentation: Each dot in a scanned Braille image is composed of light and dark areas separated by background. Therefore, the scanned image is segmented so that only three classes of regions exist: dark, light and background.

Recto dots detection: Extracts recto dots from single-sided Braille documents and recto dots from double-sided Braille documents and places them in a new image.

Verso dots detection: Extracts verso dots from double-sided Braille documents and places them in a new image.

The rest of the study is organized as follows: material and methods, results, discussion and conclusion.

MATERIALS AND METHODS

Image acquisition and understanding the braille image: The first stage of any vision system is image acquisition. After an image has been obtained, various methods of processing can be applied to it to perform many different tasks. Different image acquisition techniques have been employed by different OBR systems with each one having its own advantages and disadvantages. Flatbed scanners and digital cameras have been widely used by developers and researchers for obtaining images of Braille documents. In this study, an optical scanner was used for image acquisition. It is important to note that the resolution of the scanner should be set at 150 pixels per inch (ppi) or the measurements of Braille cells will change.

Each Braille character or "cell" is made up of 6 dots arranged in a rectangle comprising 2 columns of 3 dots each (3×2 configuration) as it can be seen in Fig. 1. The dots are conventionally numbered 1, 2 and 3 from the top of the leftward column and 4, 5 and 6 from the top of the rightward column. Each cell represents a letter, numeral or punctuation mark. A dot may be raised at any of the 6 positions, or any combination. Counting the space, in which no dot is raised, there are 64 such combinations (that is, 2 to the 6th power 2^6 = 64) (Al-Salman *et al*., 2007).

The dimensions of a Braille dot have been set according to the tactile resolution of the fingertips of a person. Dot height is approximately 0.02 inches (0.5 mm); the horizontal and vertical spacing between dot centers within a Braille cell is approximately 0.1 inches (2.5 mm); the blank space between dots on adjacent cells is approximately 0.15 inches (3.75 mm) horizontally and 0.2 inches (5.0 mm) vertically. A standard Braille page is 11 inches by 11.5 inches and typically has a maximum of 40-43 Braille cells per line and 25 lines (Al-Salman *et al*., 2007).

Inside a computer system, colored images are stored. The color image does not give information that helps in the detection process (Gonzalez and Woods, 2008). Therefore the colored scanned image is converted to gray level so that any pixel value in the image falls within the range 0-255 as could be seen in Fig. 2.

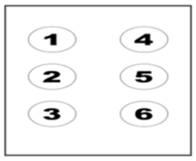


Fig. 1: Braille cell

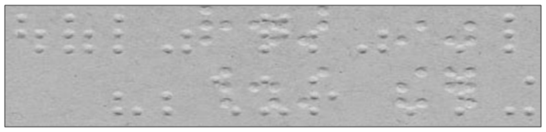


Fig. 2: A scanned Braille image converted to gra yscale

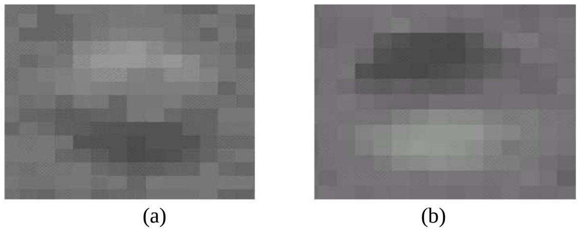


Fig. 3: An enlarged (a) recto Braille dot and (b) verso Braille dot

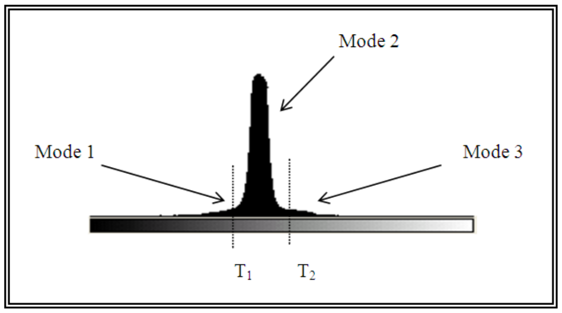


Fig. 4: Histogram of a gray scale Braille image

The scanned Braille page will have three classes of pixels: a mid-gray background, a pair of light area and dark area for each recto and verso dot. The order in which the light and dark areas appear for each dot depends upon the model of scanner used for scanning Braille documents. Some models represent recto dots (protrusions) as light areas over dark areas (Fig. 3a). In such models verso dots (depressions) are represented as dark areas over light areas (Fig. 3b). Some scanners may produce the reverse. The scanner used during our work produces the former pattern.

Braille image segmentation: As we mentioned before, a scanned Braille image will have three classes of pixels: a mid-gray background, a pair of light and dark area for each recto and verso dot. This characteristic of Braille images results in a histogram composed of three modes. Figure 4 represents the histogram of the Braille image in Fig. 2.

The three modes of a histogram of a Braille image represent the three classes of pixels:

- Mode 1: represents the dark region of a recto and verso dot
- Mode 2: represents the background
- Mode 3: represents the light region of a recto and verso dot

It should be noted that the number of thresholds is always less than the number of modes by one. Therefore in the case of a scanned Braille document, which histogram consists of three modes, the number of thresholds is two. Each threshold value falls in the valley between two modes. T_1 lies between modes 1 and 2, while T_2 lies between modes 2 and 3. Thresholding is performed on a scanned Braille image in order to segment it into three classes according to the following equation:

$$g(x,y)=\begin{cases} 0 & \text{if} \quad f(x,y) < T_1 \\ 100 & \text{if} \quad T_1 <= f(x,y) <= T_2 \\ 255 & \text{if} \quad f(x,y) > T_2 \end{cases}$$

where, g(x,y) is the segmented image, f(x,y) is the Braille image of pixel (x,y).

Thresholding is used to perform segmentation, which subdivides an image into its constituent regions or objects. During this step the calculated threshold values will be used. Each pixel in the image is checked; if its value is less than threshold value T_1 then this is a dark region pixel and will be assigned the value 0 in the output image (value 0 produces black color). If its value falls between threshold values T_1 and T_2 then this is a background pixel and will be assigned the value 100 in the output image. The last case is when the pixel value is greater than threshold value T_2 in which this is a light region pixel and will be assigned the value 255 in the output image (value 255 produces white color). The problem of thresholding is the estimation of threshold T1 and T2. In this study, we used a statistical method based on Beta distribution to estimate the thresholds. In the following, we will explain in details Beta distribution and the new stability thresholding.

Beta distribution: A statistical distribution can be characterized by two measures skewness (ß1) and kurtosis (ß2). We present a comparison among the following four distributions: Gaussian, Log-Normal, Gamma and Beta regarding those two measures. Gaussian distribution has a skewness of zero because it is symmetrical about the mean. The Log-Normal distribution is often skewed, with a slowly decreasing right tail. Both of the Beta and Gamma distributions satisfy the basic criteria of uni-modality and rightward skew, but the Gamma distribution differs from the Beta by having a shorter initial phase of low slope and a lower peak (Zaart and Ziou, 2007). Beta distribution can be skewed to left as well but Gamma cannot have the shape skewed to left. This is the main advantage of Beta over other distributions.

As shown in Fig. 5, in the space of ß1 and ß2, we can see that the Log-Normal and Gamma are represented by a line. The Gaussian distribution is represented by a point and the Beta distribution is represented by an area (El-Zaart and Ziou, 2007). From this figure, we can conclude that Beta is the best distribution for modeling a Braille image. It is the best distribution since it can be used to approximate any shape of histogram (Al-Saleh and El-Zaart, 2007).

For those reasons we have chosen to apply the Beta distribution in our method. In the following we will explain in details the Beta distribution.

The Beta distribution is a continuous probability distribution with the probability density function (pdf) defined on the interval [0, 1] Math World (El Zaart and Ziou, 2007):

$$f(x,\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\ \Gamma(\beta)}\ x^{\alpha-1}\ (1-x)^{\beta-1}$$

where, α and β are the shape parameters of the distribution and must be greater than zero. x is a random variable and it must be between 0 and 1. Γ is the gamma function. The Beta distribution can take different shapes depending on the values of its two parameters α and β as can be seen in Fig. 6. The special case of the beta distribution when α = 1 and β = 1 is the standard uniform distribution.

The moment-based estimators for α and β in a homogenous region (R_k), are given by Eq. 1 and 2:

$$\alpha = m_1^k\left(m_1^k - m_2^k\right)\Big/\left(m_2 - \left(m_1^k\right)^2\right) \quad (1)$$

And:

$$\beta = \left(m_1^k - 1\right)\left(m_2^k - m_1^k\right)\Big/\left(m_2^k - \left(m_1^k\right)^2\right) \quad (2)$$

where, the m_r^k is the r^{th} sample moment of this homogenous region (R_k) and is given by:

$$m_r^k = \sum_{i\in R_K} h(x_i)\ x_i^r \Big/ \sum_{i\in R_K} h(x_i)$$

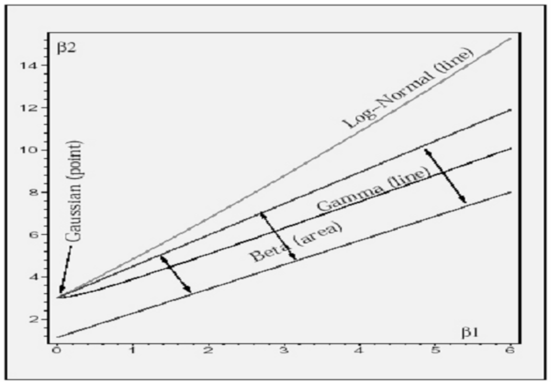


Fig. 5: Gaussian, gamma, beta and log-normal distributions (El-Zaart and Ziou, 2007)

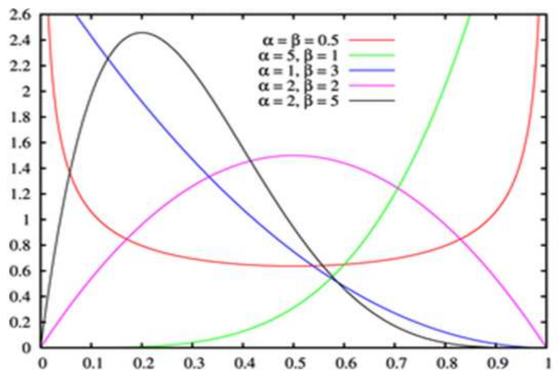


Fig. 6: Beta distribution The free encyclopedia

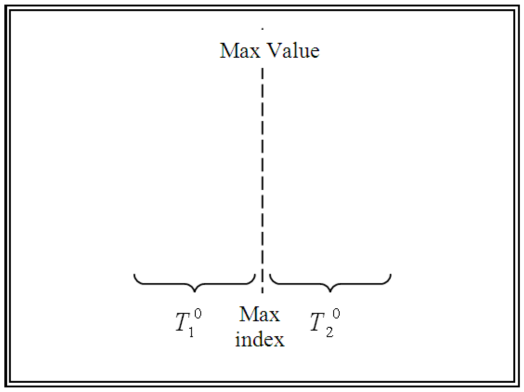


Fig. 7: Initial threshold values estimation

The Beta distribution is thus used in stability thresholding method in order to estimate the two thresholds T_1 and T_2.

New stability thresholding: In this section, we developed thresholding stability technique with Beta distribution to estimate thresholds. The process presented here is iterated through all modes of the histogram in order to estimate the two thresholds. First, the values of α, β and prior probability P of each mode of the histogram are estimated using initial values of thresholds. These alpha, beta and prior probability values are used to recalculate the thresholds. The procedure is repeated until the difference between the old and new threshold values for all modes of the histogram is equal to zero. The algorithm is summarized as follows:

- Estimate initial threshold values T_1^0 and T_2^0. To do this we first find the maximum value of the histogram and use it for the calculation of T_1^0 and T_2^0. T_1^0 will be the average gray level value of image starting from 0 to maximum value, Fig. 7. T_2^0 will be the average gray level value of image starting from maximum value to 255:

$$T_1^0 = \sum_{i=0}^{\text{MaxIndex}} (i*h(i)) \Big/ \sum_{i=0}^{\text{MaxIndex}} h(i)$$

$$T_2^0 = \sum_{i=\text{MaxIndex}}^{255} (i*h(i)) \Big/ \sum_{i=\text{MaxIndex}}^{255} h(i)$$

T_1^0 and T_2^0 are then used to estimate the values for alpha α, beta β and prior probability (P) for each mode of the histogram:

- The values of α_i and β_i for mode i (i=1,2,3) are estimated using Eq. 1 and 2
- The prior probability P for a certain mode i is estimated according to the following formula:

$$P_i = \sum_{j\in \text{Mode}\, i} h(x_j) \Big/ \sum_{j=0}^{255} h(x_j), i = 1,2,3$$

- The estimated values of α, β and prior probability for each mode are now used to recalculate the thresholds T_i^{new}, i=1,2 using the following formula:

$$T_i^{new} = 1 - e^{\frac{-A - B\log(T_i^0)}{C}}$$

Where:

$$A = \log(\frac{P_i K_i}{P_{i+1} K_{i+1}}), B = \alpha_i - \alpha_{i+1} \text{ and } C = \beta_i - \beta_{i+1}$$

$$K_r = \frac{\Gamma(\alpha_r - \beta_r)}{\Gamma(\alpha_{r+1} - \beta_{r+1})}, \qquad r = i, i+1$$

- The procedure is repeated until the error is zero. The error is calculated as the cumulative difference between the old threshold values and the new threshold values as the following equation:

$$|\,(\text{ error } = (T_1^0 - T_1^{New}) - (T_2^0 - T_2^{New})\,|$$

Otherwise, we set $T_i^{New} = T_i^0$ (i=1,2) and repeat steps 2 until 4.

By the end of this algorithm the optimal values of T_1^{New} and T_2^{New} are found.

After calculating threshold values, we then segment the Braille image. The segmented image is thus used to detect all Braille dots.

Recto dots detection using grid: The procedure presented in this here is used for detecting recto dots from double-sided Braille documents. In order to accomplish this task, a grid is formed first and then detection of dots takes place. At the end we handle the problems we faced during recto dots detections.

Grid formation: The segmented image produces three classes of pixels: dark, gray (background) and light. Using the segmented image, a grid is formed by first selecting a starting point and then generating horizontal lines with regular intervals and vertical lines. The starting point is selected to be just above the first dot in the first cell of a scanned Braille document. Using Segmented image, the original Braille image is scanned line by line until reaching a light region that might be part of a recto dot. A check is made to ensure there exists a dark region below this detected light region. This is done to guarantee that this is a recto dot. When such a dot is found, it is considered to be the first recto dot in the first line of a scanned Braille document.

Horizontal lines generation is straightforward being compared with vertical lines generation. This is due to the regular horizontal arrangement of Braille characters. However this is not the case with the vertical arrangements of characters. This is because, as in natural language, a line contains words of different character lengths. Therefore, we cannot predict the spaces between characters in the same line and draw vertical lines accordingly. Using a segmented image as a reference helped greatly for the generation of vertical lines as we will see later. In the followings, we will discuss horizontal and vertical lines generation.

Horizontal lines generation: The procedure for horizontal lines generation goes as follows. As we mentioned, the first horizontal line is drawn above the first line of characters of a scanned Braille document. The location of this line is determined using segmented image.

It should be noted that the spacing between dots within the same Braille cell is the same (indicated by arcs number 2 in Fig. 8). The spacing between Braille cells is also the same (indicated by arcs number 1 in Fig. 8). The space between characters is larger than that between dots within the same character. According to this information, horizontal lines are generated.

In all cases, the line location is kept to help for the generation of vertical lines. The procedure goes on until all horizontal lines are generated as in Fig. 8.

Vertical lines generation: After all horizontal lines have been generated, the procedure starts generating vertical lines. To do this, only blocks that are expected to hold recto dots (indicated by arcs number 2 in Fig. 8) are checked. The algorithm starts here by identifying light regions using the segmented image. It starts from the first horizontal block (formed by two lines) and it is applied to all blocks in the image as follows:

- The current block is checked for having any recto dots. This is done by finding light regions in this block
- If a recto dot is found then a vertical line is drawn before this dot and another line after it. The two vertical lines start from the top of the scanned Braille document until the bottom
- If this is the end of the block then the algorithm stops. If not, then go to step 1

At the end, each recto dot will be bounded in a box. The result of this step can be seen in Fig. 9.

It is important to note that for the first step in the previous algorithm we have to make sure that a detected light region is not part of a bottom region of a verso dot. So a check is made on this region to see if there exists a dark region above it. At the end of this test, if this is a verso dot which was mistakenly recognized as recto one, then it will not be counted as a recto dot.

As well for the second step, a test is established to ensure that no vertical line has been drawn before or after a detected light region. If it was found that no lines have been already drawn then they will be drawn as indicated in the second step.

Dot detection using grid: After a gird with each of the recto dots contained within a box has been formed, detection should then take place. The detection process goes as follows. Starting from the first horizontal block in the image resulted from the previous step, only boxes of a certain size range that are expected to hold recto dots are checked Fig. 10.

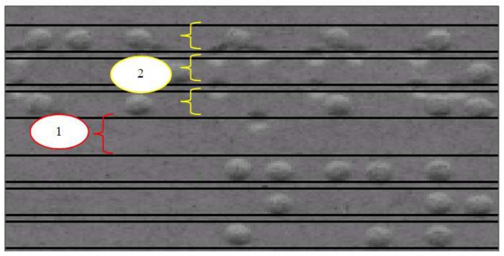


Fig. 8: Horizontal line generation for forming a grid for recto dots detection

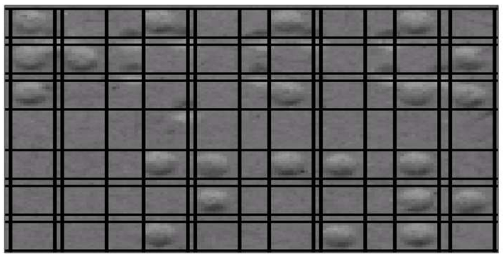


Fig. 9: Grid formation for recto dots detection

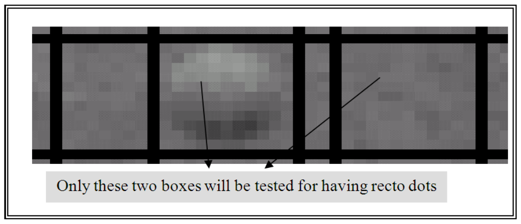
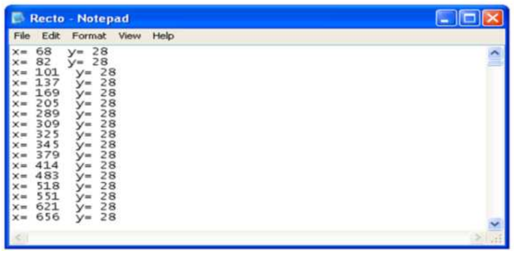


Fig. 10: Recto dot detection



```
x= 68  y= 28
x= 82  y= 28
x= 101  y= 28
x= 137  y= 28
x= 169  y= 28
x= 205  y= 28
x= 289  y= 28
x= 309  y= 28
x= 325  y= 28
x= 345  y= 28
x= 379  y= 28
x= 414  y= 28
x= 483  y= 28
x= 518  y= 28
x= 551  y= 28
x= 621  y= 28
x= 656  y= 28
```

Fig. 11: A text file that contains the coordinates of the centers of recto dots

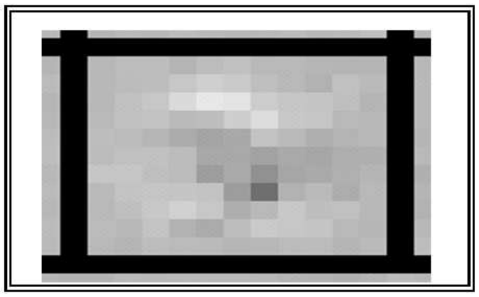


Fig. 12: A recto dot with no clear separation of light and dark regions

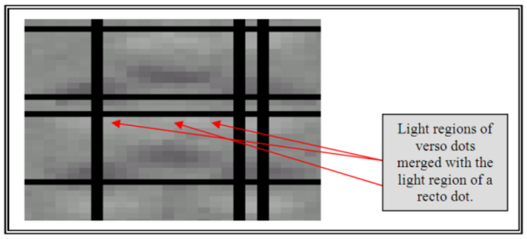

Light regions of verso dots merged with the light region of a recto dot.

Fig. 13: White regions of verso and recto dots merged

For each box in the grid, a test is carried out to decide whether it holds a recto dot or not. This test works both horizontally and vertically. It checks if the upper half contains a light region and the lower half contains a dark region. If this was the case then this is considered a recto dot and will be drawn on the output image in the same location. The output of this step is a text file that contains the coordinates of the center of each found recto dot (Fig. 11).

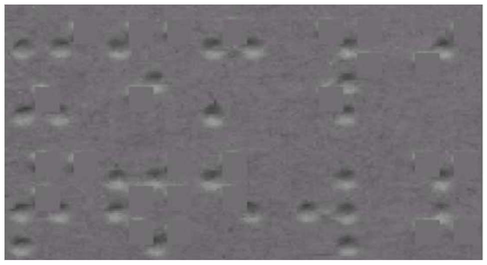


Fig. 14: Part of a scanned braille document with recto dots covered with boxes

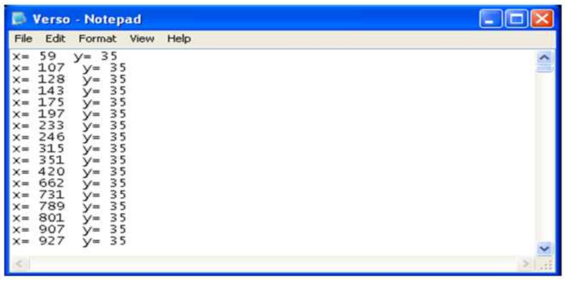
Problems encountered in recto dots detection: We have faced many problems in the detection of recto dots. In the following, we will discuss the most important ones.

Some recto dots did not have clear separation between their light and dark areas. This has made detection of such dots harder. An example can be seen in Fig. 12. However, horizontal and vertical checking worked very well with such situations.

The dark and the light regions are not usually connected; they are separated by background pixels. However, it is not uncommon for regions of the same type to be connected. This situation happens only in double-sided documents, in which case the light region, for instance, of a recto dot is merged with the light region of a verso dot. This is one of the major problems encountered when attempting to identify dots forming pairs Fig. 13. By using a grid we could overcome this problem because each recto dot is contained within a box by itself.

Verso dots detection: The process followed for detecting verso dots from double-sided Braille documents is similar to that for detecting recto dots. The only difference is that processing was carried on the original image in the case of recto dots detection. Next, we will work on a modified image in the case of verso dots detection.

The procedure goes as follows. Using the text file produced from the Recto Dots Detection step, we delete all recto dots from the original image. This is done by creating an image with all recto dots covered with boxes Fig. 14. The gray color used for covering detected recto dots is chosen to be in the midway between the two threshold values T_1 and T_2. It is important for this value to be chosen so that it does not fall within the range for dark or light pixels, i.e., less than T_1 or greater than T_2 respectively, but rather in the range of background pixels. This will prevent pixels of those boxes from being detected as parts of light or dark regions of verso dots.



```
x= 59  y= 35
x= 107  y= 35
x= 128  y= 35
x= 143  y= 35
x= 175  y= 35
x= 197  y= 35
x= 233  y= 35
x= 246  y= 35
x= 315  y= 35
x= 351  y= 35
x= 420  y= 35
x= 662  y= 35
x= 731  y= 35
x= 789  y= 35
x= 801  y= 35
x= 907  y= 35
x= 927  y= 35
```

Fig. 15: A text file that contains the coordinates of the centers of verso dots

Afterwards the work conducted before is repeated here for verso dots detection. Figure 15 contain the coordinates of the center of each found verso dot.

RESULTS

The proposed method has been tested with a wide variety of scanned Braille documents, with different colors, both single and double sided, written in the Arabic language. Three different flatbed scanners were used. It should be noted that when applying our method on brown Braille documents, no filter was needed. However, due to the light and hard to detect reflections caused by white and yellow Braille documents, we had to use a filter. The general recognition results given here refer to the detection of dots composing characters.

Preliminary results from images of both single-sided and double-sided Braille documents seem very promising. Overall, on single-sided and double-sided documents with average defects, all dots were correctly recognized. However, images with low and very low quality were not tested.

In the case of double-sided documents, the additional problem of the merged dot components (light and/or dark regions) is present as a potential cause of missed dots. But hopefully our system is precise and flexible enough to detect all dots correctly. To put these results into perspective, for documents scanned using a standard scanner, all fall within the average difficulty category, the system achieved perfect results.

In the following, we will show the results of applying our method on both single-sided and double-sided documents.

Single-sided scanned braille document: The original image of a single-sided scanned Braille document is illustrated in Fig. 16.

Table 1 illustrates the initial and estimated threshold values resulted from applying our method on the image in Fig. 16. It took 6 iterations in order to reach those final threshold values.

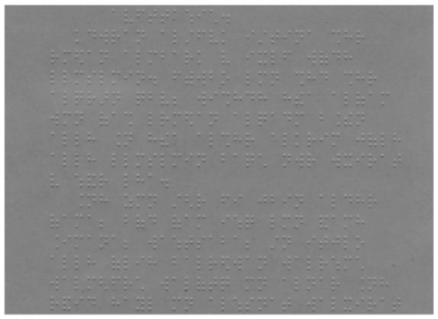


Fig. 16: Single-sided scanned braille document

Table 1: Initial and estimated threshold values for the image in Fig. 16

	T1	T2
Initial value	126	135
Estimated value	119	142

Table 2: Initial and estimated values for alpha, beta and prior probability resulted from applying the proposed method on the image in Fig. 16

	Mode 1		Mode 2		Mode 3	
	Initial value	Final value	Initial value	Final value	Initial value	Final value
α	104.85000	84.5877	1590.64000	590.34	218.93600	188.22
β	119.46000	116.2100	1519.05000	560.53	182.87300	129.50
P	0.101492	0.0800	0.724848	0.80	0.17366	0.12

Table 3: Initial and estimated threshold values for the image in Fig. 20

	T1	T2
Initial value	106	117
Estimated value	99	125

The image resulted from applying segmentation to the image in Fig. 16 can be seen in Fig. 17.

As well, the initial and estimated values for alpha, beta and prior probability for each of the three modes of the histogram can be seen in Table 2.

Part of the image resulted from applying the grid on the image in Fig. 16 can be seen in Fig. 18.

And finally the image containing detected recto dots from the image in Fig. 16 can be seen in Fig. 19.

Double-sided scanned braille document: The original image of a double-sided scanned Braille document is illustrated in Fig. 20.

Table 3 illustrates the initial and estimated threshold values resulted from applying our method on the image in Fig. 20. It took 8 iterations in order to reach those final threshold values.

The image resulted from applying segmentation to the image in Fig. 20 can be seen in Fig. 21.

As well, the initial and estimated values for alpha, beta and prior probability for each of three modes of the histogram can be seen in Table 4.

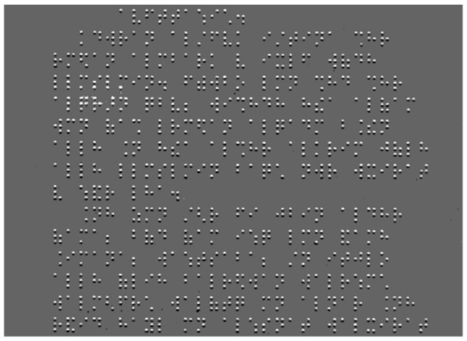


Fig. 17: The segmented image of Fig. 16

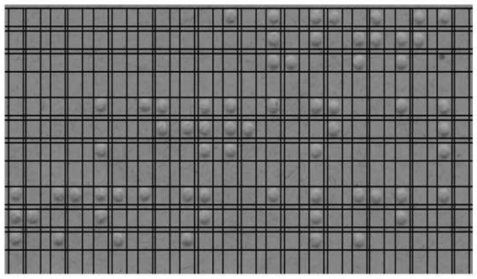


Fig. 18: Grid applied to the image in Fig. 16

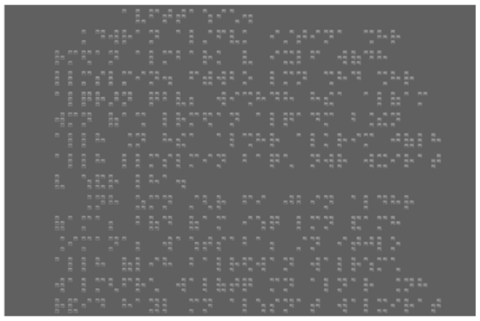


Fig. 19: Dots detected from the image in Fig. 16

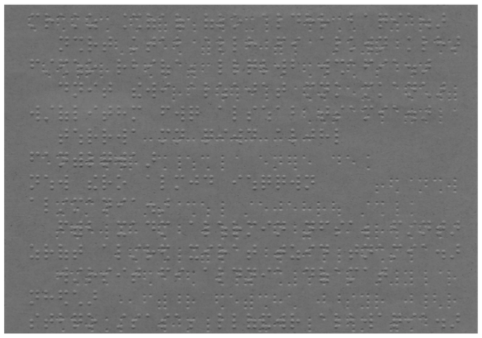


Fig. 20: Double-sided scanned braille document

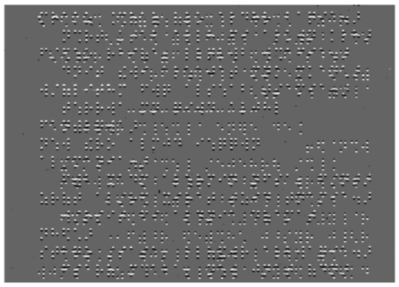


Fig. 21: The segmented image of Fig. 20

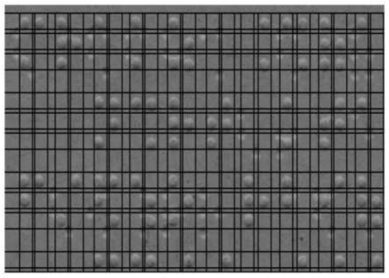


Fig. 22: Grid applied to the image in Fig. 20

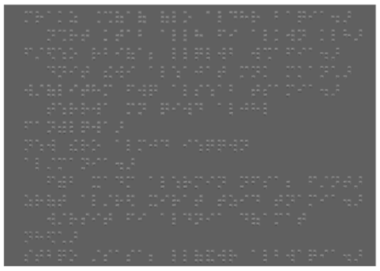


Fig. 23: Recto dots detected from the image in Fig. 20

Table 4: Initial and estimated values for alpha, beta and prior probability resulted from applying the proposed method on the image in Fig. 20

	Mode 1		Mode 2		Mode 3	
	Initial value	Final value	Initial value	Final value	Initial value	Final value
α	64.778	70.8109	939.582	378.885	102.279	137.182
β	105.912	135.5660	1206.230	483.054	109.115	120.273
P	0.110	0.0800	0.720	0.810	0.170	0.110

For double-sided scanned Braille documents we will first present recto dots extraction results and then verso dots extraction results.

Recto dots extraction: Part of the image resulted from applying the grid on the image in Fig. 20 for detecting recto dots can be seen in Fig. 22.

The image containing detected recto dots from the image in Fig. 20 can be seen in Fig. 23.

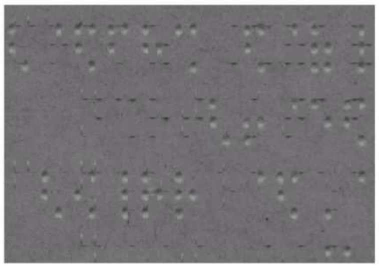


Fig. 24: Recto dots in the image in Fig. 20 covered with mid-gray color boxes

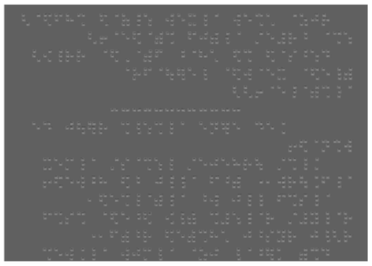


Fig. 25:Verso dots dected from the image in Fig. 24

Verso dots extraction: Figure 24 shows part of the image resulted from covering all recto dots in the original image by boxes.

The image in Fig. 25 contains detected verso dots from the image in Fig. 24.

DISCUSSION

The proposed algorithm has been applied on single-sided Braille documents. From Fig. 16-19, the result of dots detection was good and all the dots are detected. The algorithm also applied on double-sided Braille documents (Fig. 20-24). The results of recto and verso dots detection was good. The proposed algorithm is based on the fact that the scanned Braille documents are well scanned without rotation. Otherwise, the vertical and horizontal lines detection algorithm does not work well and consequently we will have missing recto and verso dots.

CONCLUSION

This study described a new algorithm to detect dots in image of embossed Braille material obtained by an optical scanner. Although the Braille dots have the same color as the background, they cast soft shadows when scanned with a standard flatbed scanner. These shadows are used to locate the dots on the page. As the dots on both sides of the page are visible from one side, both sides of the page can be recognized in a single scan.

We applied a new stability thresholding with Beta distribution in order to initiate the process of a multi-mode estimator that calculates threshold values. Segmentation is then performed using those values. To ensure correct detection of dots composing Braille characters, a grid is formed. By using a grid, many detection problems were eliminated; most importantly is the problem caused by the merging of light or dark regions between dots.

As a whole, the approach described here shows the feasibility of a cost-effective, fast and easy method to detect dots in Braille documents. It does not require expensive or complicated hardware. It uses a flatbed scanner which can be shared with other applications. Robustness to cope with low quality scans and defective documents is built-in at different levels, from the initial thresholding through to the flexible Braille point grid construction.

Future work is focused on converting the dots to Braille cells and then to text by producing translation rules for the conversion of Braille into print for the corresponding natural language. Expanding the translation capabilities to different types of Braille should also be taken into account.

The implication of documents with different colors in the analysis is also under study. The elimination of using a filter with both white and yellow Braille documents should be studied. In the case of skewed images, some mechanisms to handle rotation should also be implemented.

ACKNOWLEDGMENT

This stuyd was supported by the National Plan for Science and technology, King Saud University, Riyadh, Saudi Arabia.

REFRENCES

Al-Saleh, A. and A. El-Zaart, 2007. Unsupervised learning technique for skin images segmentation using a mixture of beta distributions. Proceeding of the 3rd Kuala Lumpur National Conference on Biomedical Engineering (KLNCBE'06), Malaysia, pp: 304-307. DOI: 10.1007/978-3-540-68017-8_78

Al-Salman, A.M., Y. Al-Ohali, M. Al-Kanhal and A. Al-Rajih, 2007. An Arabic optical braille recognition system. Proceedings of the 1st International Conference on ICT Accessibility, Apr. 12-14, Hammamet, Tunisia, pp: 81-87.

Antonacopoulos, A. and D. Bridson, 2004. A robust braille recognition system. Lecture Notes Comput. Sci., 3163: 111-114. DOI: 10.1007/978-3-540-28640-0_50

Braille Institute, 2011. Empowering visually impaired people to live fulfilling lives.

Bunke, H. and A.L. Spitz, 2006. Document Analysis Systems. 1st Edn., Springer, Berlin, New Yor, ISBN 3540321403, pp: 630.

El-Zaart, A. and D. Ziou, 2007. Statistical modeling of SAR images. Int. J. Remote Sens., 28: 2277-2294. DOI: 10.1080/01431160600933997

Gonzalez, R.C. and R.E. Woods, 2008. Digital Image Processing. 3rd Edn., Prentice Hall, Upper Saddle River, NJ., ISBN: 013168728X, pp: 954.

Hedgpeth, T., M. Rush, V. Iyer, J.A., Black, M., Donderler and S., Panchanathan, 2003. iCare-reader-a truly portable reading device for the blind. Proceedings of the 6th Accessing Higher Ground Conference: Assistive Technology and Accessible Media in Higher Education, University of Boulder, Boulder, CO USA.

Mennens, J., L.V., Tichelen, G., François and J.J., Engelen, 1994. Optical recognition of braille writing using standard equipment. IEEE Trans. Rehabilitation Eng., 2: 207-212. DOI: 10.1109/86.340878

Mihara, Y., A. Sugimoto, E. Shibayama and S. Takahashi, 2005. An interactive braille-recognition system for the visually impaired based on a portable camera. CHI Extended Abstracts, pp: 1653-1656. DOI: 10.1145/1056808.1056989