Braille Recognition System – With a Case Study Arabic Braille Documents

Rawan Ismail Zaghloul
Department of management Information Systems, Al-Balqa'a Applied University, Amman College, Amman-Jordan
E-mail: rawanzaghloul@yahoo.com
P.O. Box: 923199; Postal Code: 11192

Tomader Jameel Bani-Ata
Department of management Information Systems, Al-Balqa'a Applied University, Amman College, Amman-Jordan
E-mail: tom_bani_ata@yahoo.com
P.O. Box: 8604; Postal Code: 11121

Abstract

According to the World Health Organization, about 314 million people are visually impaired, with 45 million blind [1]. As a minority, they are often overlooked when it comes to design new systems. This is why there should be greater emphasis on developing the suitable means to help them communicating with their world.

Advancement in technology should be employed to help those individuals reading Braille documents and communicating with sighted people easily. Thus, the development of an efficient Optical Braille Recognition (OBR) system becomes an essential approach to be considered. Many of the proposed systems are time consuming and labour intensive. This paper introduces a new OBR system which designed for recognizing a scanned Arabic Braille document and converting it into a computerized textual form that could be utilized by converting it into voice using other applications, or it could be stored for later use. System integration appears from the starting stage of pre-processing up to the cell detection and interpretation stages. The proposed system is not only possesses excellent detection rates up to 99% with average processing time around 25 second per page, but also, it could be applied over any Braille document regardless to the writing grade or language.

Keywords: Braille, Braille recognition system, OBR, pattern recognition, binary image, Arabic alphabet, morphological operations, image processing.

1. Introduction

In the daily life there are many cases in which it is important for visually impaired or blind people to communicate with sighted people. Braille system is the most famous method that enables visually impaired people to create documents which could be read through touch [2].

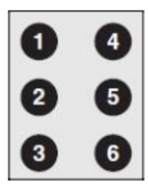
Many computerized systems were proposed to translate scanned Braille documents into a textual form in such language. This paper introduces an efficient Optical Braille Recognition (OBR) System where Arabic Braille documents are scanned and converted into Arabic text. The proposed system is composed of the pre-processing, the cell detection and the interpretation stages respectively.

2. Braille System

Braille system is a method that enables blind or visually impaired people to write and read through the concept of Braille cell. The cell consists of a series of raised dots that can be read with their fingers. The dots are created by embossing on a thick sheet using a manual machine or a special printer. [3]

Each Braille cell or character contains six dots positions that are arranged and numbered as shown in Figure 1. A dot may be embossed or raised at any of the six positions. So, there are 64 combinations that can be represented using the six positions. It is essential that the dots are uniform in size and position to let the reader understand the characters easily. There are different ways for writing Braille documents such as, writing in grade one where a cell can be used to represent an alphabet letter, a number, a punctuation mark, or the space symbol. Or writing in grade two where a cell may represent a contraction of a word, or even a whole word. [4, 5]

Figure 1: Braille cell or character



3. OBR Systems

OBR system is a software that reads scanned Braille documents and converts them into digital text files. These files can be stored for later use, or can be processed immediately in other applications such as converting textual files into voice to help blind people understanding them easily [6, 7].

Recently, OBR systems become very useful especially for Braille users and for everyone who wants to communicate with blind people and cannot read Braille characters such as teachers, parents, friends, computerized Braille libraries, and public organizations. Furthermore, these systems enable blind people to read Braille documents easily, and to save them in a small storage space with a low cost [4].

Actually, many efforts were focused on developing OBR systems such as the algorithm that is proposed by Al-Salman and others for Braille cells recognition of double sided Braille documents, through the segmentation of the Braille image using a stability thresholding method with a mixture of Beta distributions [8]. While Al-Shamma and others present a system for design and implementation of Optical Arabic Braille Recognition with voice and text conversion for single sided Braille documents [9]. However, in 2009, Abdelmonem and others developed an OBR system that is completely invariant to scale of the scanned image. Their technique can be applied regardless the grade or the language of Braille documents [5]. Furthermore, in 2004, Lisa Wong and others built a software solution prototype to optically recognize single sided Braille documents using neural networks [10].

4. Methodology

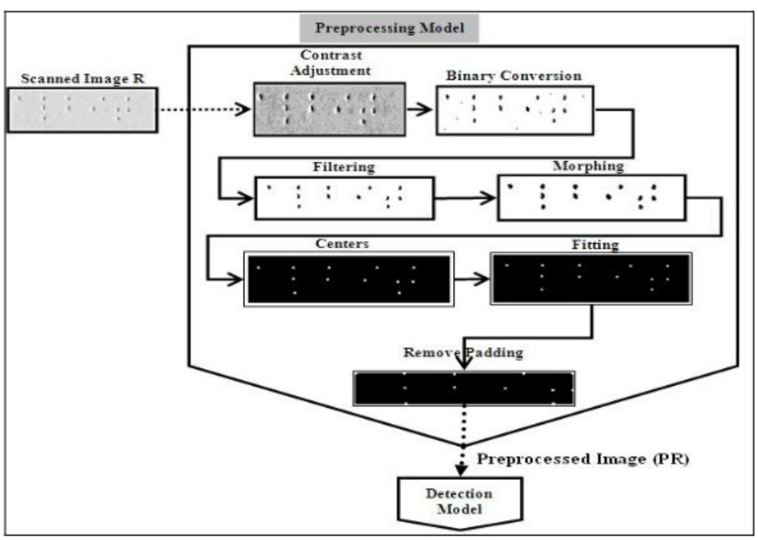
This section introduces the proposed system, which composed of three main stages: the pre-processing, the cell detection and the interpretation stage respectively.

4.1 Pre-Processing Stage

This stage started by importing a scanned Braille image *R* in order to be processed as shown in Figure 2. At first, you have to increase the contrast of the image by mapping the intensity values in *R* to new values, such that 1% of data is saturated at low and high intensities of *R* [11, 12], then the image will be converted into binary format. As a result of the contrast adjustment and the binary conversion processes, the binary image *R* will be affected by noise. Thus, after experiments, the median filter is elected to de-noise the image. Afterwards, the enhancement of dots shape is necessary. Therefore, the use of morphological operations -*series of erosion and dilation operations using the disk shape*- is suggested.

Then it is suggested to compute the central point for each segment (Braille dot) in the image. But actually, this process led consequently to produce inaccurate positions for some centers in the image. In order to solve this problem, the centers must be fitted horizontally and vertically according to a threshold value. Hence, for each column contains centers, try to find the neighbor columns in the same region (± threshold value) and move all centers in these columns to one column, then repeat the same process over the whole rows in the image. Finally, remove zero-padding from the fitted image. Now, the pre-processed image (*PR*) is ready to pass through the cell detection stage.

Figure 2: Pre-processing Stage



4.2 Cell Detection

The proposed cell detection model consists of three processes: Finding the distance vector, Building the Matching Board, and finally the Matching process.

4.2.1 Process A: Finding the Distance vector

In this process, the Find_Distance_Vector algorithm is proposed to determine the vector D which consists of: The vertical distance between two points within a cell (VD), the horizontal distance between two points within a cell (HD), the vertical distance between two adjacent cells (DBV), and the horizontal distance between two adjacent cells (DBH) as shown in Figure 3.

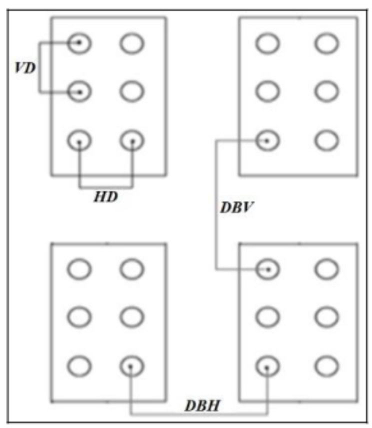
The find-distance-vector-algorithm attempts to compute HD and VD by obtaining the minimum horizontal distance ($minHd$) and minimum vertical distance ($minVd$) along the whole image respectively. While, it computes DBV and DBH as depicted in equations (1 & 2):

$$DBV = 2.4 \times VD \quad (1)$$

$$DBH = 1.4 \times HD \quad (2)$$

Where, the values 2.4 and 1.4 are the estimated vertical and horizontal ratios. These ratios have been obtained after several experiments on Braille documents, depending on the linear proportion between cell distances in a Braille cell.

Figure 3: Distances within and between cells



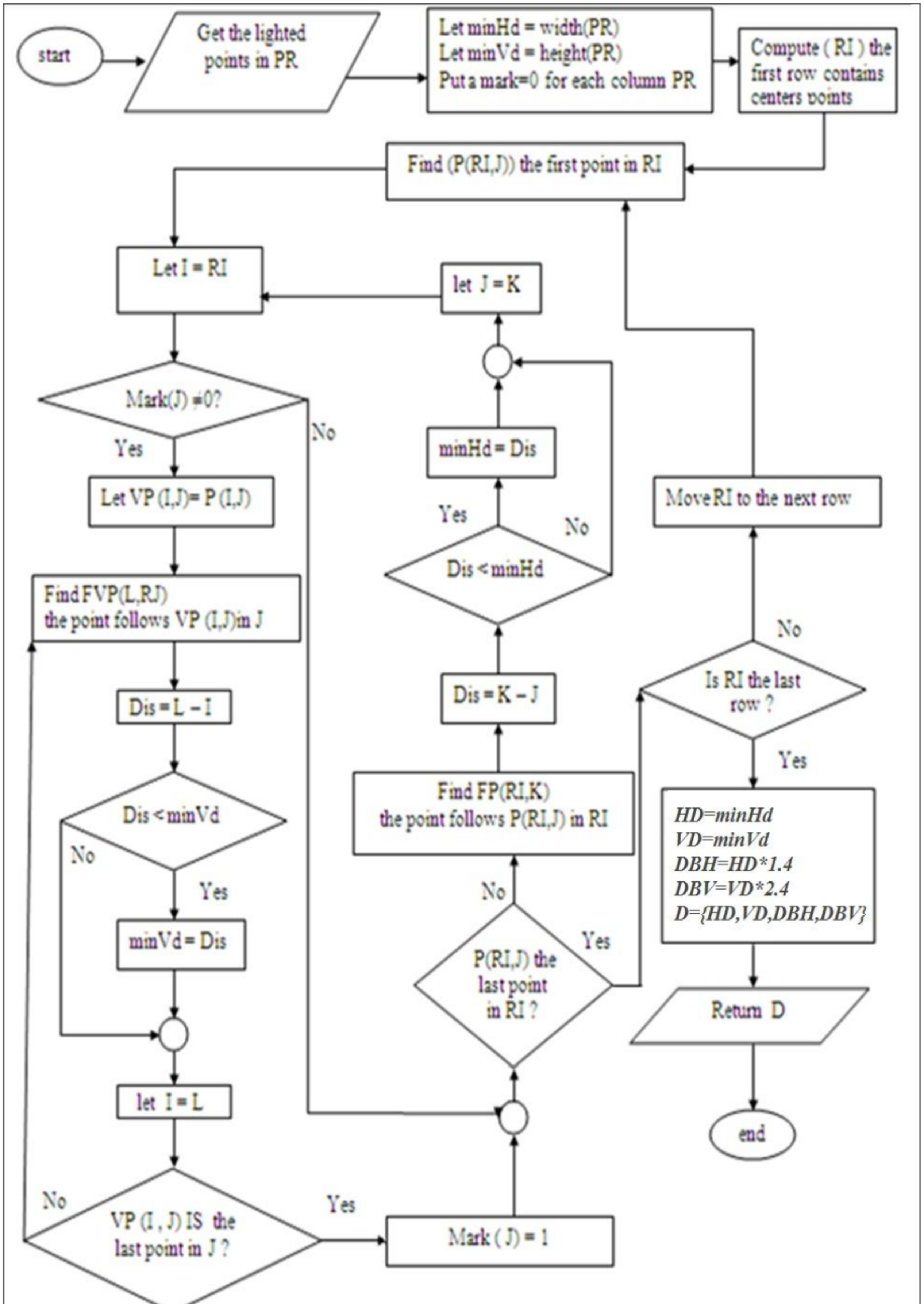
The general layout for the proposed algorithm is as follows:

For each visited row in the scanning process of the image rows, do the following:

1. Find the first point (P) in the row.
2. Find the vertical minimum distance according to the column of P if and only if this column is not visited before. Then mark the column to ensure that this column is visited; this is to skip it later when the scan visits a new point in another row but in the same column.
3. If P is not the last point in the row, find the horizontal minimum distance according to the next point in the same row. Then, move P to the next point, and repeat from process 2. But, if P is the last point in the row, move to the next row and repeat from process 1.

As illustrated in figure 4, the image will be scanned vertically through the horizontal scan of the image in order to find $minVd$ and $minHd$. For example, if the first row contains points in all possible columns, then the vertical minimum distance will be obtained only after finishing the scan of the first row. That's because all columns become visited.

Figure 4: The flowchart of Find_Distance_Vector Algorithm



4.2.2 Process B: Building a Matching Board

As a result of the fitting process in the pre-processing stage, cells sizes in *PR* may vary, as shown in Figure 5. This affects the accuracy of the obtained result from the Find_Distance_Vector algorithm. This problem can be solved by creating a *Matching Board* (*M*) for the image *PR* to be used in the next process for justifying dots positions in *PR*. The created *M* must be built depending on the vector *D* and the size of *PR*. Thus, *M* is an image with the same size as *PR* that contains the exact locations of all possible dots. Figure 6 shows a sample image with its created matching board.

The matching board *M* for *PR* is created by firstly computing the standard locations for all dots positions in the first cell *C* using equation 3.

$$C = \begin{cases} P(r,c) \\ P(r+Vd,c) \\ P(r+2*Vd,c) \\ P(r,c+Hd) \\ P(r+Vd,c+Hd) \\ P(r+2*Vd,c+Hd) \end{cases} \tag{3}$$

Where, *P(r,c)* is the first point in C that is located at row *r* and column *c*. The other points in equation 3 are presented in the order of their corresponding positions in the standard Braille cell as shown in Figure 1.

After drawing the first cell, the process will be continued to draw the horizontal and vertical adjacent cells until the size of *M* becomes equal to the size of *PR*. Notice that, the movement to the standard adjacent cells horizontally or vertically is done according to the calculated values of *DBH* and *DBV*. On other words, the movement is performed by calculating the position of the first point in the horizontally adjacent cell as depicted in equation 4, while the vertical movement is applied by calculating the position of the first point in the vertically adjacent cell as depicted in equation 5.

$$P(r,c) \leftarrow P(r,c+HD+DBH) \tag{4}$$

$$P(r,c) \leftarrow P(r + 2 * VD + DBV,\ c) \tag{5}$$

Figure 5: An example of cells sizes variation as a result of fitting process

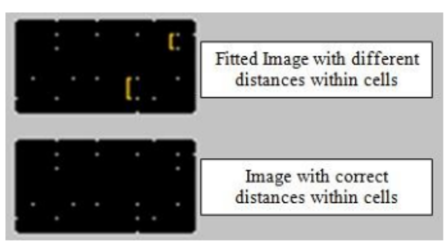
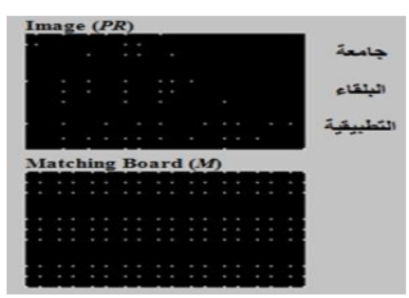


Figure 6: Matching Board for an image



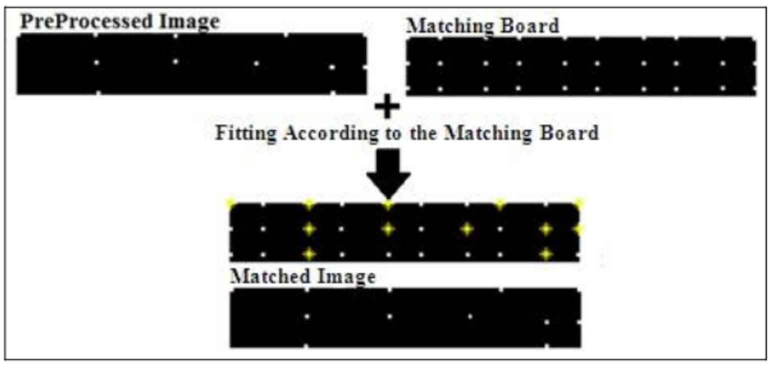
4.2.3 Process C: Matching

This process aims to rebuild the pre-processed image *PR* according to its matching board *M* by locating each point in *PR* on its correct position after matching it with the nearest point in *M*. Figure 7 shows an example of the matching process. The matching algorithm is as illustrated here:

For each point *P(r,c)* in *PR*: if its position is not correct according to M, move it to the nearest position as follows:

- *IF P(r,c)* is located in a wrong row, THEN check its column as follows:
  - IF it is correct, MOVE *r* to the nearest row in *M*.
  - IF it is wrong, MOVE *r* to the nearest row in *M* and MOVE *c* to the nearest column in *M*.
- *IF P(r,c)* is located in a correct row, THEN check its column as follows:
  - IF it is wrong, MOVE c to the nearest column in *M*.
  - IF it is correct, so it is located in the correct position.

Figure 7: Matching example

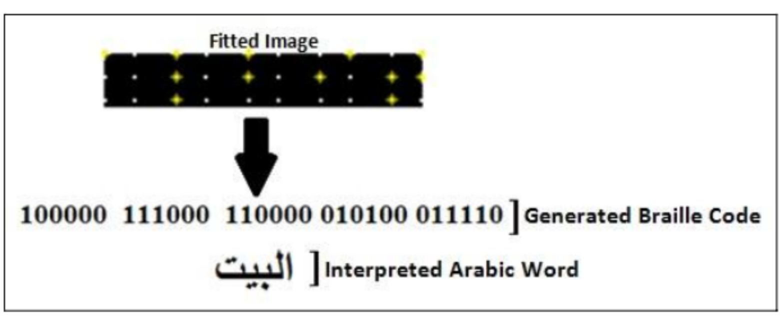


4.3 Interpretation

At this stage, a database is created for all Arabic Braille cells with their corresponding Binary codes. The binary code of a cell consists of six binary bits; 1's for the positions of lighted points in the cell, and 0's for the other positions. Each cell in the matched image will be converted into its binary code. Then, the generated code will be interpreted to the corresponding Arabic character or phrase according to the stored database. An example for interpretation is shown in Figure 8.

Actually, upgrading this system for working over any language is extremely simple, by connecting the interpreter with the database codes for the desired language.

Figure 8: Interpretation example



5. Experimental Results

To test and evaluate the performance of the proposed system, the experiments are applied over a large dataset of scanned single sided Arabic Braille documents with different scanning resolutions. Some of these documents were imported from Braille printed documents (Arabic Braille books or magazines), and the others were imported from manually written documents. A recognition rate of 99% is achieved over the printed documents, while 97% is achieved over manually written documents.

As performance measures, the processing time (*PT*) of the *OBR* system was calculated. To obtain results that are as consistent as possible, the average of the results was taken. The system achieved an average *PT* around 25 second per page. All experiments were performed under the Matlab environment on a *PC 3 GHz Pentium IV*, with *1GB* of memory.

The overall error rate of the proposed system can be attributed to the quality of the acquired image of the Braille document. Such errors may be caused by: The presence of dark areas that are not parts of Braille dots, or the presence of a valid dot outside the expected size of the Braille cell.

6. Conclusions

In this paper, a new OBR system is proposed for scanned single sided Braille documents, with special reference to the Arabic Braille documents. The proposed system shows high performance and excellent detection rates for different image resolutions. Furthermore, upgrading this system is extremely simple, by connecting the interpreter with the database codes for the desired language in such grade.

References

[1] Y. Bar-Coh en, "Electroactive polymers for refreshable Braille displays", SPIE Newsroom, doi: 10.1117/2.1200909.1738.

[2] P. Daniels, 1996. "Analog and Digital Writing", The World's Writing Systems, Oxford University Press, New York, Oxford, pp. 886-892.

[3] A. Bhattacharjee, A. Ye, J. Lisak, M. Vargas, and D. Goldreich, 2010. "Vibrotactile Masking Experiments Reveal Accelerated Somatosensory Processing in Congenitally Blind Braille Readers", The Journal of Neuroscience, Vol 30 pp.14288-14298, doi:10.1523/JNEUROSCI.1447-10.2010.

[4] A. Al-Salman, Y. AlOhali, M. AlKanhal, and A. AlRajih, 2007. "An Arabic Optical Braille Recognition System" in Proceedings of the First International Conference in Information and Communication Technology & Accessibility ICTA'07, Tunisia, pp. 81-86.

[5] M. Abdelmonem, M. El-Hoseiny, A. Ali, K. Emara, H. Abdel Hafez, A. Gamal, 2009. "Dynamic Optical Braille Recognition (OBR) System" in Proceedings of the International Conference on Image Processing, Computer Vision, & Pattern Recognition IPCV'09, Las Vegas Nevada, USA, pp. 779-786.

[6] Neovision s.r.o, 2004. "Optical Braille Recognition System Version 3.7 User manual".

[7] L. Jie; Y. Xiaoguang; Z. Dayong, 2010. "Optical Braille recognition with Haar wavelet features and Support-Vector Machine" in Computer, Mechatronics, Control and Electronic Engineering International Conference (CMCE'10), Changchun, pp. 64 – 67, doi: 10.1109/CMCE.2010.5610062.

[8] A. Al-Salman, A. El-Zaart, A. Al-Suhaibani, Y. Al-Hokail, K. Al-Qabbany, A.-A.O, 2010. "An Efficient Braille Cells Recognition" in the 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM'10), Chengdu, pp. 1-4, doi: 10.1109/WICOM.2010.5601020.

[9] S. Al-Shamma and S. Fathi, 2010. "Arabic Braille Recognition and Transcription into Text and Voice" in the 5th Cairo International Biomedical Engineering Conference, Cairo, Egypt, pp.227-231.

[10] L. Wong, W. Abdulla, S. Hussmann , 2004. "A Software Algorithm Prototype for Optical Recognition of Embossed Braille" in International Conference on Pattern Recognition - ICPR '04, Cambridge, England, UK, Vol. 2, pp.586-589.

[11] M. Sathik, S. Sujatha, 2011. " Wavelet Based Blind Technique by Espousing Hankel Matrix for Robust Watermarking", International Journal of Advanced Science and Technology, Vol. 26, pp. 57-72, ISSN:2005-4238.

[12] M.Sathik1 and S.Sujatha, 2010. "An Improved Invisible Watermarking Technique for Image Authentication", International Journal of Advanced Science and Technology, Vol. 24, pp. 61-74.