DESIGN AND IMPLEMENTATION OF A NEW PERSIAN DIGITS OCR ALGORITHM ON FPGA CHIPS

Navid Toosizadeh and Mohammad Eshghi

Electrical and Computer Engineering Department, Shahid Beheshti University
Evin, 1983963113, Tehran, Iran
phone: + (98) 29901, fax: + (98) 21241-7940, email: n-toosizadeh@std.sbu.ac.ir, m-eshghi@sbu.ac.ir
web: www.sbu.ac.ir

ABSTRACT

The method presented in this paper is mostly based on recognition using vertical and horizontal projections of Persian digits, along with other characteristics of them. The presented method makes recognition of Persian digits fast with 0% error. Since, at this stage the main goal is designing an algorithm suitable for hardware implementation, a single size font is considered. To implement the algorithm of Persian digits OCR, VHDL codes were used, compiled and fitted in a proper FPGA.

Keywords: Persian OCR, VHDL, FPGA, Optical Character Recognition, Projection.

1. INTRODUCTION

Traditionally, Persian Optical Character Recognition (OCR) algorithms are developed and implemented in the software media. With growing applications of FPGA chips, implementing a Persian OCR algorithm on this hardware is desired. Portability, affordability and speed of optical character recognition implemented on the hardware are among the main goals for hardware implementation of an OCR algorithm on FPGA chips.

In the recent years, many researchers conducted in order to invent effective algorithms for Optical Character Recognition (OCR) of Persian and Arabic texts [1-7]. However, with growing applications of Field Programmable Gate Arrays (FPGAs), no effort is done to implement these algorithms on hardware chips. In this paper, the results of implementation of a Persian OCR algorithm on FPGA chips solely to recognize Persian digits are presented.

The algorithm presented in [1] is based on Fourier transform of characters. Recognition based on morphology is proposed in [2], where algorithm of [3] is based on projections. In some other efforts, researches conduct to improve the results of Persian OCR outputs using MAP statistical modelling [4].

In the next section, a new algorithm suitable for hardware implementation is described. Section 3 contains the details of implementing the algorithm. The results of implementing are presented in section 4. Conclusion and references are followed in the next sections.

2. ALGORITHMS OF PRESENTED OCR

The block diagram of a general OCR system is shown in figure 1. Figure (1-a) depicts the training phase, and figure (1-b) shows the tracking phase of the OCR.

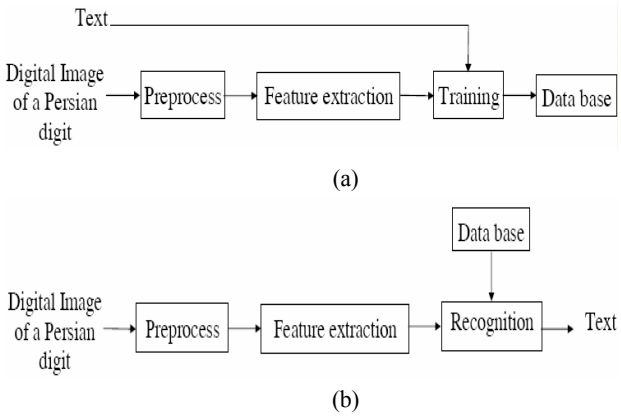


Figure1. A general OCR system, (a) Training phase, (b) Tracking phase

2.1 Training

The training phase of the OCR is used to make a data base which is used in the recognition phase of the OCR.

The desired samples of different digits with the predefined font and size are applied to the system. After the feature extraction, these features are used to build the data base.

2.1.1 Digitization

For the functional simulation, Persian digits were first typed using Microsoft Word. Then bitmap of these digits were produced using a 300 dpi scanner.

Windowing around each digit is done separately. This windowing can be done easily since digits are not connected. A conventional method is based on projections of the line of digits and finding the space between them by verifying these projections.

A matrix of 10 × 7 pixels can completely show all of these digits with a size of 14 in the bitmap format.

2.1.2 Preprocess

It must be emphasized that pixels of body of a digit are assumed to be shown with '0' and the background pixels with '1'.

In this stage, a procedure was used which looks for zeros of bitmap that do not belong to the digit, i.e. small regions not connected to the main body of the digit. The fact that Persian digits do not contain dots shows these zeros are noise. In other words, every small region of the text that is not connected to the main body is a noise region and must be cancelled.

2.1.3 *Feature extraction*

Four specifications of Persian digits' image in the presented method are defined as:

MH: Maximum number of zero pixels in the horizontal projection

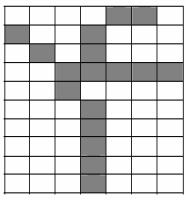
MV: Maximum number of zero pixels in the Vertical projection

TP: Total number of zero Pixels of the image

TLQ: Number of zero pixels in the Top Left Quarter of the image

These four specifications of all ten Persian digits are determined. As an example, the specifications of digit "4" with font size 14 and font type "Tahoma" are shown in Table 1. In this table the column "Shape" shows the way by which a digit can be shown in a 10× 7 matrix.

Table 1. Four specifications for Persian digit "4"

Persian Digits		Features			
Value	Shape	MH	MV	TP	TLQ
Four		8	5	17	3

The features of all Persian digits with font size 14 and font type "Tahoma" are presented in table 2.

Table 2. Four specifications for all Persian digits

Persian Digits		Features			
Value	Shape	MH	MV	TP	TLQ
Zero	۰	4	3	8	0
One	۱	6	1	9	3
Two	۲	6	5	15	6
Three	۳	6	4	15	6
Four	۴	8	5	17	3
Five	۵	4	4	20	3
Six	۶	3	3	12	2
Seven	۷	3	2	16	4
Eight	۸	3	2	16	2
Nine	۹	6	4	17	5

2.1.4 Data Base

As mentioned in section 2.1.3, the features of all Persian digits are extracted. These features are used to build a data base which is used in the recognition phase. This data base for font "Tahoma" with font size 14 is shown in table 2.

2.2 Tracking

The tracking phase consists of four parts as Preprocess, Feature Extraction, Data Base and Recognition. The Preprocess, Feature Extraction and Data Base parts are exactly similar to those of training phase. The recognition part is described in the next section.

2.2.1 Recognition

The digital image of each digit is entered into the tracking phase. After preprocessing of the image, the four features of MH, MV, TP and TLQ of the image are extracted. The three features, MH, MV and TP, are compared to the features of each digit i in the data base. This comparison is made according to equation 1 in which,

DbMH(i) is the Maximum number of zero pixels in the Horizontal projection of the compared digit i in the data base,

DbMV(i) is the Maximum number of zero pixels in the Vertical projection of the compared digit i in the data base,

DbTP(i) is the Total number of zero Pixels in the compared digit i in the data base and

Devn(i) is the difference between the features of the input digit and the features of digit i.

$$Devn(i)=|MH-dbMH(i)|+|MV-dbMV(i)|+|TP-dbTP(i)| \quad (1)$$
$$(\text{for } i=0, \ldots, 9)$$

In a sequential processing method, the minimum of Devn's is calculated and saved in a temporary variable named "Devo". In each stage if "Devn" and "Devo" are equal, the process of recognition decides due to the fourth feature of TLQ. This feature is mostly used to distinct between digit "7" and digit "8".

Finally, the digit i that has the minimum deviation or difference with the input digit is selected as the output digit.

By changing the input text font and size in the training phase and thus changing the data base, new fonts with different sizes could be supported. This could be accomplished by using other images for different digits. In other words this implementation is independent of the text font and size.

3. IMPLEMENTATION

The implementation of each block shown in figure 1 using VHDL code development and FPGA implementation is described in this section.

3.1 Scanner and preprocessing blocks

A separate hardware may be used to handle the scanner output, window the digits and feed them to the training and tracking systems. The outputs of this system are saved in 10 × 7 matrices for the font size 14 of "Tahoma" Persian font.

These bitmaps were also used in the test bench for testing the general system.
A VHDL type for matrix of digits bitmap named "bit_10by7" was defined. This type shows the 10 × 7 frame of digits with size of 14 (figure 3).

```
type bit_10by7 is array(0 to 9,0 to 6) of bit;
```

Figure 2. VHDL defined type "bit_10by7"

The "Preprocess" subsystem is implemented using a procedure named "Preprocess" which is included in the package containing all necessary procedures for Persian OCR.

3.2 Feature extraction block

Entity that does the process of feature extraction is named "feature_extractor". This block is combined of two primary subcomponents named "projection2features" and "bitmap2projections". The block diagram is shown in figure 3.

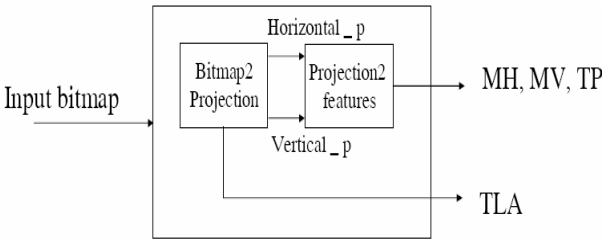


Figure 3. Feature extractor block diagram

3.2.1 "Bitmap2projections" Entity

This module receives the bitmap from the input and computes its horizontal and vertical projections. It uses a procedure that calculates the number of foreground pixels (i.e. the number of zero bits in the image of the digit) on every column of the image. This vector is named "Horizontal_p". This procedure also calculates the number of zero bits on every row of the image of the digit and that is "Vertical_p" vector. Therefore, "Horizontal_p" has 7 elements and "Vertical_p" has 10 elements for font "Tahoma" with size "14". These two matrices determine the horizontal and vertical projections of the input digit.
Additionally, "bitmap2projections" computes the number of foreground pixels of the digital image in the top left quarter of it. This property is named TLQ and is obtained by calculating the number of zero bits in the top left quarter of the figure. To do this, "bitmap2projections" uses a procedure named "qsummer". This procedure returns the number of zeros in the top left quarter of the figure due to the number of column and row of each bit.

3.2.2 "Projection2features" Entity

This module computes MH, MV, and TP features according to the horizontal projection (Horizontal_p) and vertical projection (Vertical_p) from the "bitmap2projections" block. "Projection2features" benefits a procedure named "projection2features".

To calculate TP, "Projection2features" block adds up all the elements of "Horizontal_p" or "Vertical_p" vectors. The way to compute MH and MV is comparing each element of "Horizontal_p" or "Vertical_p" with the maximum of the previous elements; if the present element is greater than the preceding maximum, it replaces the maximum.

3.3 Recognition block

The entity that does the recognition of different digits is named "Recognition". This entity utilizes a procedure named "Recognition". The procedure compares the collected features (MH, MV, TP and TLA) with the data base. A part of the VHDL code for the recognition procedure is shown in figure 4. In this code, "DbTLQ(i)" is the number of zero pixels in the top left quarter of digit i. Also, "dif_TLQ" is a variable containing the difference between TLQ of the input digit and TLQ of the digit i in the data. "Output_number" is the output of the procedure containing the recognized number in the proper form.

```
Begin

  while (i/=10) loop
        devn:=abs(MH-DbMH(i))+abs(MV-
        DbMV(i))+abs(TP-DbTP(i));
        if Devn<Devo then
                Devo:=Devn;
                num:=i;
        elsif Devn=Devo then
          if abs(TLQ-DbTLQ(i))<dif_TLQ then
        num:=i;
          end if;
        end if;
        dif_TLQ:= abs(TLQ-DbTLQ(i));
        i:=i+1;
  end loop;
output_number<=numbers(num);

end recognize;
```

Figure 4. A part of VHDL code of Recognition procedure

To implement the necessary data base, some constants were used, which are used to present four features, MH, MV, TP and TLQ of each digit i.

4. RESULTS

Simulation results and implementation are presented below.

4.1 Recognition block

Simulation of "Persian digits OCR" was done with the aid of "ModelSim" software. The results were satisfactory. For example if noise is limited to 1 or 2 pixels, no errors will occur in recognition. Only when recognition of digit 8, if the noise level in the top left quarter region of the figure is high, the recognition system may mistake the input number with digit 7. For different digits, 100 samples were used to train

the system and 100 other samples were used to test it. The total error is 0 percent for different samples of the predefined font and size.

4.2 Implementation results

VHDL codes were synthesized with Leonardo 2003a software for a FPGA of "ALTERA" company. The selected family is "Stratix". The total number of used logic cells is 870 and the maximum clock frequency for inputting rows of the input image matrix is 21 MHz for the C5 device. In other words, the delay between the entrance of input data and readiness of output is 47 nanoseconds for the C5 device. The selected FPGA is EP1S10F484C5. 8.22% of the total logic cells of this FPGA are used.

5. CONCLUSION

An algorithm and a hardware system for a fast and accurate optical recognition of Persian digits were presented. The system combines a number of modules for which, VHDL codes were developed. The algorithm was implemented on a special FPGA family from Altera Company, namely Strarix. The design can be easily migrated into other similar FPGAs. The font size and type is flexible. Different font types and sizes could be used to train the system.

The hardware resources used for this system are small when compared to available resources for modern FPGAs (8.22% of the total logic cells of Stratix10). The speed for this real time system is high (21 MHz for processing each row of the digit matrix) and could be improved using faster hardware methods, i.e. ASIC.

6. REFRENCES

[1] R.Safabakhsh and V.Pakdast, "Structural recognition of hand script using morphology method," Third Annual CSI computer CSICC'97, Tehran, Iran, 1997.

[2] R.Safabakhsh and V.Pakdast, "Recognition of Persian characters using projection," Third Annual CSI computer CSICC'97, Tehran, Iran, 1997.

[3] E.Kabir, K.Bahariand and M.Ahmadzadeh, "Recognition of typed Persian text," Tarbiyat Modares Uni., Tehran, Iran, 1996.

[4] R.Mehran, A.Shali and F.Razzazi, "A statistical correction-rejection strategy for OCR outputs in Persian personal information forms, ICITA, 2004.

[5] M.Eshghi, "Transformation of Persian computer printed text to Braille," Shahid Beheshti University, Tehran, Iran, 2000.

[6] M.Kavianifar and A.Amin, "Preprocessing and Structural Feature Extraction for a Multi-Fonts Arabic / Persian OCR," New South Wales university, Sydney, Australia, 2000.

[7] R.Schwartz, C.LaPre, J.Makhoul, C.Raphael and Y.Zhao, "Language-Independent OCR Using a Continuous Speech Recognition System," Proc. IEEE ICPR, USA, pp. 99-103, 1996.