Design of Filter for Fingerprint Application on FPGA
S Gayathri¹ Tanveerahamad Rozewale²

¹,²Department of Electronics and Communication Engineering
¹,²SJCE, Mysuru, India

Abstract—This paper describes the design of filter for fingerprint application using parallel architecture which makes fast processing, reduces the processing time, increases speed and will make it useful for biometric authentication. This paper also describes about the comparison of MATLAB TO FPGA Implementation and presents a set of efficient criteria for implementing fingerprint image on FPGA Virtex5 board with optimization in device utilization.

Key words: Fingerprint; Preprocessing; Parallel architecture; Filtering, Device utilization

I. INTRODUCTION

Every human being’s fingerprint is unique. Each fingerprint has a unique characteristic and pattern that is made up of lines and spaces. The lines are called ridges while the gap between the ridges are called valleys. The significances about using fingerprint biometrics is that more widely accepted, reliable and convenient than other forms of physical identification in case of privacy is considered.

The fingerprint image recognition process involves preprocessing, feature extraction and post-processing [5] the preprocessing involves the normalization, orientation, filtering, binarization and thinning image. Normalization: The gray levels of the input image is compared with that of standard values, normalize the pixel values of the image if the gray levels of the image are not in the standard range. The input image, which is to be processed, is first stored inside a ROM as a text file. This image pixels mean is calculated using Accumulator and divider blocks. Then the mean used to compute the variance of the image. Then mean and variance outputs of the image applied to the Comparator and Adder/Subtractor block to get the Normalized image. Orientation: The orientation field of a fingerprint image defines the local orientation of the ridges contained in the fingerprint. Distinct ridges provide accurate extraction of the minutiae from the fingerprint images. Binarization: Compares each pixel to particular threshold which then changes its value to either pure white (0x00) or pure black (0xFF). The threshold used is local block mean. Thinning: Thinning process is used to skeletonize the binary image by decreasing all lines to a individual pixel thickness. Filtering: This is a mechanical process of manually finding and removing ridge structures that probably do not belong and by filling in gaps in ridges that do not belong. The reason for using Gabor filter is because of its characteristics spatial localization, orientation selectivity and spatial frequency selectivity.

This paper shows the filtering of generic image or also data base image where filtering is the one of the major step involved in fingerprint recognition process in preprocessing steps. The basic idea of fingerprint recognition process can be explained by following figure.

II. RELATED WORK

The objective of design and simulation of gabor filter using verilog code[1] in fingerprint recognition is to segment the texture of fingerprint. This includes the design of a single filter to segment multiple (> 2) textures and the design of multiple filters to segment multiple textures. The computational efficiency becomes increasingly important as the size of images, number of Gabor prefilters, number of Gaussian postfilters, and number of textures increases. This work has completed only with simulation, in contrast to its future scope the current paper gives well solution for it.

The use of gabor filters in the paper[2] is 16 which presents matching scheme that utilizes a ridge feature map to match fingerprint images. The technique described here obviates the whose spatial frequencies correspond to the average inter-ridge spacing in fingerprints, is used to capture the ridge strength at equally spaced orientations. The
genuine accept rate of the Gabor filter based matcher is observed to be ~ 10% to 15% higher than that of minutiae-based matcher at low false accept rates. Fingerprint feature extraction and matching takes nearly 7.1 seconds on a Pentium IV, 2.4 GHz processor. However we are presenting paper which works on the latest Virtex5 board with speed in terms of 1us/1ps for the simulation of filtering.

The implementation of orientation block [3] has been done in Virtex2 pro with better performance but the future scope remained there with to implement the Filtering block on FPGA. This current paper implements the filtering block on Virtex5 FPGA with low speed grade of -2.

In this work[4], the Log-Gabor filter is singularity in the log function at the origin, the spatial domain is not accurately constructed so the filters are constructed in the frequency domain. This work shows the construction with help of the Fourier transform (FT) for the whole image and implement the filtering with a bank of Log-Gabor filters in frequency domain.

The horizontal mask of vertical detection is as shown in figure 3.2. It calculates the difference among the pixel intensities of a particular edge. As the center row of mask consists of zeros it does not include original values of edge in the image but rather it calculates the difference between above and below pixel intensities of particular edge. Thus increase the sudden change of intensities making the edge more visible.

The implementation of edge detection in parallel architecture is as shown in figure 3.2. It calculates the difference of right and left pixel values around that edge. Also the center values of both first and third column is 2 and -2 respectively. This gives more weightage to the pixel values around the edge region. This increases the edge intensity and it becomes enhanced compared to original image.

<table>
<thead>
<tr>
<th>i11</th>
<th>i11</th>
<th>i12</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>i11</td>
<td>i12</td>
</tr>
<tr>
<td>i21</td>
<td>i21</td>
<td>i22</td>
</tr>
</tbody>
</table>

Fig. 3.1: shows 3x3 image

In parallel architecture the filter block of pre-processing stage is implemented using symmetric extension method, to filter out noise present in the fingerprint image. Symmetric extension is useful in reducing the boundary distortion. Symmetric extension is accomplished by reflecting the rows and columns at the image boundary. The image is divided into 8 X 8 matrix and symmetric extension is applied, resulting in 10 X 10 matrix. This 10 X 10 matrix is applied to a processing element which is nothing but a filter.

The processing element makes use of Sobel filter to remove the noise in the oriented image. The filtered output is stored in the RAM for further processing. Since parallel architecture as shown in figure is employed, the computation required for filtering is ¼ th of that of serial architecture. The filtered output is same in both the cases but only the resource utilization differs. Parallel architecture requires more hardware in comparison with serial architecture. Sobel operator is used for edge detection Vertical Detection and Horizontal Detection. The vertical mask of vertical detection as shown in Table 3.1. When mask is applied on the image, it works like a first order derivate and calculates the difference of pixel intensities in pixel edge region. As the center column consists of zero it does not include the original values of an image but instead it calculates the difference of right and left pixel values around that edge. Also the center values of both first and third column is 2 and -2 respectively. This gives more weightage to the pixel values around the edge region. This increases the edge intensity and it becomes enhanced compared to original image.

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>x3</th>
</tr>
</thead>
<tbody>
<tr>
<td>x4</td>
<td>x5</td>
<td>x6</td>
</tr>
<tr>
<td>x7</td>
<td>x8</td>
<td>x9</td>
</tr>
<tr>
<td>y1</td>
<td>y2</td>
<td>y3</td>
</tr>
<tr>
<td>y4</td>
<td>y5</td>
<td>y6</td>
</tr>
<tr>
<td>y7</td>
<td>y8</td>
<td>y9</td>
</tr>
</tbody>
</table>

Fig. 3.2: shows sobel values X and Y

The implementation of edge detection in parallel architecture is as shown in figure 3.2. It calculates the difference among the pixel intensities of a particular edge. As the center row of mask consists of zeros it does not include original values of edge in the image but rather it calculates the difference between above and below pixel intensities of particular edge. Thus increase the sudden change of intensities making the edge more visible.

In symmetric extension method, 10x10 image there is no loss of original data. The image coefficients are multiplied with Sobel X values giving the Gabor coefficients.

**B. Design:**

A Gabor filter is linear filter whose impulse response is defined by a harmonic function multiplied by Gaussian.
function. The Fourier transform of a Gabor filter’s impulse response is the convolution of Fourier transform of harmonic function and the Fourier function of Gaussian function.

\[ g(x, y) = s(x, y) \cdot wr(x, y) \]  

(1)

where \( s(x, y) \) is a complex sinusoidal, known as the carrier, and \( wr(x, y) \) is a 2-D Gaussian-shaped function, known as the envelop. The general form of filter [1] using Gabor function can be represent as

\[ G(x, y, \theta, f_0) = \exp \left[ -\frac{1}{2} \left( \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right) \right] \cos (2\pi f_0 x \cos \theta + y \sin \theta) \]

(2)

where \( \theta \) is the ridge orientation with respect to vertical axis, \( f_0 \) is the selected ridge frequency in \( x_\theta \)-direction, \( \sigma_x \) and \( \sigma_y \) are the standard deviation of Gaussian function along the \( x_\theta \) and \( y_\theta \) axes respectively and the \( [x_\theta, y_\theta] \) are the coordination of \( [x, y] \) after a clockwise rotation of the Cartesian axes by an angle of \( (90-\theta) \) [8].

By tuning a filter to specific frequency and direction, the local frequency and orientation information can be obtained.

IV. IMPLEMENTATION

Implementation of Filter: Consider the flow chart of parallel architecture to implement Gabor filter on FPGA

- The Fingerprint image is presented in matrix form or in pixel.
- The orientated image pixels are convolved with coefficient values in MAC unit (convolution between image pixels with kernel values).
- The control unit takes the data from memory location and sent it to the multiplication-accumulator (MAC) unit.
- In MAC, ROM stores the coefficient or kernel value of the filter.
- When both input data and kernel values are available multiplication and accumulation would take place in the MAC.
- The result of MAC is the filtered image.

The proposed work is to implement Gabor filter based on memory base architecture for real-time convolution with variable kernels. Firstly store the input data (image pixel) in the memory. The size of the memory depends on the pixel size. If the image is of size 16x16 then the memory size will be 16x16. It means that every memory location will store a value for 1 image pixel. Then start the convolution process. The control unit takes the data from memory location and sent it to the multiplication-accumulator (MAC) unit. In MAC, ROM stores the coefficient or kernel value of the Gabor filter. When both input data and kernel values are available multiplication and accumulation would take place in the MAC. The result of MAC is the filtered image. The flowchart of Gabor filter is shown in Figure 3.4. Input Memory block stores the incoming signal in the form of image pixel in the memory. The kernel value defines the ridge and valley region of fingerprint. MAC block is used to perform the convolution operation. The convoluted signal with the Gabor coefficient is transformed into a matrix format. Simulation of the filter is carried out using verilog HDL language using Xilinx software and synthesized code is generated. The simulation will generate the schematic according to the code.

The flow of the convolution process between image pixels and coefficient kernels [5] is as follows. Method of convolution and the sequence of memory reading are as shown in Figure 4.2 and Figure 4.3. Firstly the pixel \( D_{11} \) is convoluted with the kernel. The value of \( D_{11} \) after convolution is given by

\[ D_{11} = (D_{00} \cdot W_{11}) + (D_{01} \cdot W_{12}) + (D_{02} \cdot W_{13}) + (D_{10} \cdot W_{21}) + (D_{11} \cdot W_{22}) + (D_{12} \cdot W_{23}) + (D_{20} \cdot W_{31}) + (D_{21} \cdot W_{32}) + (D_{22} \cdot W_{33}) \]  

(3)
Comparison of MATLAB and FPGA Implementation.

MATLAB Implementation: The MATLAB is a high level language which is here a software reference model for hardware implementation. In this MATLAB code a resized 128x128 image for which we have 0 to 255 gray scale level.

Fig. 4.4: Resized image 128x128 which is divided into 16x16 cells that is 256 cell which further divided into 8x8 image for parallel processing of image. This is an 8bit processing such that will have four cells of each 8x8 image processing, here again for filtering 3x3 matrix of the pixel values which multiplies with coefficients matrix (3x3)(sobel values) gives us filtered value such that n terms after matrix multiplication will have n+1 terms in matrix multiplication as shown in the figure 4.2 and 4.3.

Fig. 4.5: shows 128X128 image broken down to 8x8 cells totally 16 X 16 = 256 cells

Fig. 4.6: shows Individual Cell

Fig. 4.7: shows {16,16} Here the MATLAB code is written in such fashion to get the output of gabor filter image, noise removal image. Considering generic image/database image it has to convert it into text file then have to call as penguins data in verilog code implementation.

FPGA Implementation:

Filter block will be design and implement Filter block is implemented using Gabor function on FPGA, Verilog code simulation using Xilinx software and synthesized code is generated. The simulation will generate the schematic according to the code. Then generating programming file and configuring target device with analyzing the device the Program running is done.

V. EXPERIMENTAL RESULTS

MATLAB results: The MATLAB code is performed in MATLAB2013 version.

Fig. 5.1: a Input image

Fig. 5.1: b Scale invariant features

Fig. 5.1: c Orientation filtering
Design of Filter for Fingerprint Application on FPGA

Fig. 5.1: d Gabor Filter output image
Fig. 5.1: a,b,c,d shows MATLAB output figures files
FPGA results: This Implementation is performed with Xilinx ISE tool Here four output indicates the parallel processing of 8x8 image simultaneously

Fig. 5.2: RTL Schematic
The top_split block is constituted after adding sources to the top_split block program which can be seen as follows in figure 5.3.

Fig. 5.3: RTL schematic of internal of the top_split block

Device Utilization summary:

<table>
<thead>
<tr>
<th>Device Utilization Summary (estimated blocks)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic block</td>
<td>Used</td>
<td>Available</td>
</tr>
<tr>
<td>Number of IOB Expresses</td>
<td>302</td>
<td>482</td>
</tr>
<tr>
<td>Number of Black Cells</td>
<td>846</td>
<td>1,210</td>
</tr>
<tr>
<td>Number of LUTs</td>
<td>355</td>
<td>574</td>
</tr>
<tr>
<td>Number of flip-flops</td>
<td>367</td>
<td>444</td>
</tr>
<tr>
<td>Number of 18K bit RAMs</td>
<td>13</td>
<td>18</td>
</tr>
<tr>
<td>Number of DSPs</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Fig. 5.4: shows the device utilization and output simulation of Gabor filter
The following figure shows RTL schematic of final chip block which is obtained after making master ucf file

Fig. 5.5: Shows the RTL schematic of final chip block
FPGA virtex LX110T ML505 model

Fig. 5.6: Implementation after configuring the target device on FPGA Virtex5 XUPLEX505T
VI. CONCLUSION AND FUTURE SCOPE

The input image either from generic/database image is filtered using gabor function in MATLAB which is served as a reference software for hardware implementation on FPGA using Hardware Description Language Verilog code. The image is converted into text file (penguins data) which is filtered using verilog code and finally implemented on FPGA. The filtering is done with 8 bit which can be further modified. Filtering is implemented using Gabor function because of orientation selectivity, so one can perform this by using other techniques[6] and even ASIC implementation[7] in future.

REFERENCES

[1] S. Gayathri Research scholar, SJCE, Mysore, Karnataka, India, Dr. V. Sridhar, Principal PECSE, Mandya, Karnataka, India “Design and simulation of Gabor filter using verilog HDL”.

[2] Muhammad Umer Munir and Dr. Muhammad Younas Javed College of Electrical and Mechanical Engineering, National University of Sciences and Technology, Rawalpindi, Pakistan “Fingerprint Matching using Gabor Filters”.


[5] Feng Zhao*, Xiaou Tang Department of Information Engineering, The Chinese University of Hong Kong, Shatin, NT, Hong Kong Received 4 June 2005; received in revised form 17 June 2006; accepted 6 September 2006” “Preprocessing and postprocessing for skeleton-based fingerprint minutiae extraction”.


[8] S. Gayathri, V. Sridhar Dept of E&C, Principal SJCE, Mysore, Karnataka, India PESCE, Mandya, Karnataka, India “Image enhancement technique for fingerprint recognition process”.

Fig. 5.7: shows successful run of program in device5