Biometric embedded system architecture for hand vein Identification system on FPGA

Submitted by

TALHA QAYYUM

MASTERS

IN

ROBOTICS AND INTELLIGENT MACHINE ENGINEERING (RIME)

YEAR

2016

THESIS SUPERVISOR

DR. SYED OMER GILLANI

SCHOOL OF MECHANICAL AND MANUFACTURING ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

Table of Contents

DEDICATION

Dedicated to Allah (s.w.t), my Family and beloved wife

DECLARATION

We hereby declare that no portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this of any other university or other institute of learning. In any act of plagiarism found, we are fully responsible for every disciplinary action taken against us depending upon the seriousness of the proven offence, even the cancellation of our degree

COPY RIGHT STATEMENT

- Copyright in a test of this thesis rests with the student author. Copies (by any process) either in full, or of extracts, may be only in accordance with the instructions given by author and lodged in the Library of SMME, NUST. Details may be obtained from the librarian. This page must be part of any such copies made. Further copies (by any process) of copies made in accordance with such instructions may not be made without the permission (in writing) of the author.
- The ownership of any intellectual property rights which may be described in this thesis is vested in SMME, NUST, subject to any prior agreement to the contrary, and may not be made available for use by third parties without the written permission of SMME, NUST which will describe the terms and conditions of any such agreement.
- Further information on the conditions under which disclosure and exploitation may take place is available from the library of SMME, NUST, and Islamabad.

ACKNOWLEDGMENTS

It is the only Allah Almighty who is the source of every knowledge. Thanks Allah Almighty that gave me power and determination to complete this research project. A special thanks to my beloved parents and respected teachers that have been a permanent source of encouragement and motivation for me. I am also thankful to my colleagues that have supported me whenever I needed their help.

Abstract

Biometric authentication is using world widely for different applications in security, attendance metering and for surveillance. Hand vein biometric system is mostly not used widely due to limited resources of machine. In order to overcome resource-constraints. This paper proposes novel method to implement canny edge detection algorithm for extraction vein patterns using CoreGen. The proposed algorithm is implemented on Field programmable Gated Array (FPGA) device. The proposed algorithm improves the False Acceptance Rate (FAR) as compared to existing algorithm while recognition speed is also measured.

This is a novel implementation of a canny edge detector that takes advantage of 4-pixel parallel computation. It is a pipelined architecture that uses on-chip BRAM memories to cache data between the different stages. The exploitation of both hardware parallelism and pipelining creates a very efficient design that has the same memory requirements as a design without parallelism in pixel computation. This results in achieving increased throughputs for high resolution images and a computation time of 3.09ms for a 1.2Mpixel image on a Spartan-6 FPGA. We present synthesis and simulation results for low-end and high-end Xilinx FPGAs and have achieved higher speeds, better throughputs and efficiency than the implementations presented above.

List of Figures & Tables

List of Abbreviations

OCR	Optical Character Recognition
ROS	Robot Operating System
NMS	Non-Maximum Seperation
TH	Threshold Hysteresis

Chapter 1

Introduction

1. Introduction to Topic

Verification is enter in our data society. Keeping in mind the end goal to get to administrations, resources, physical areas or data a choice is expected to whether a subject is approved to do as such. It is not achievable to physically perceive and validate people in substantial scale, mechanized frameworks. The entrenched strategies for programmed confirmation in light of information and ownership are being tested amid the most recent decades by biometric frameworks. The principle contrast lies in the bijective connection between electronic identifier and physical personality. This prompts to a few intriguing properties like non-revocation, trouble of replication, robbery and misfortune.

Biometrics offer incredible focal points over conventional confirmation strategies, however the connection between advanced representation and physiological or behavioral body properties challenges protection. The potential for abuse is natural. Crooks can utilize it for wholesale fraud and profiling, governments can utilize the innovation for controlling the populace. In this manner exceptional care must be taken when planning frameworks utilizing biometric information. By and large crude information contains therapeutic data: the information itself must be dealt with as private and profoundly touchy. Therefore we proliferate the joining of security elements and protection insurance as ahead of schedule as conceivable amid the plan period of the applications and the biometric pipelines.

These days biometric frameworks are normally in view of unique mark or 2d-confront data mostly because of verifiable, money related and client comfort contemplations. In any case, the sensors of these frameworks can as a rule be effectively evaded with fake ancient rarities; liveness and fake identification are not unimportant. Moreover the biometric data must be viewed as open, since face pictures can be effortlessly obtained on a separation if

not accessible in the apparently non-deteriorating "recollections" of the Internet. Fingerprints then again are left accidentally on surfaces of items all through that we touch in regular day to day existence. It is a minor undertaking, and generally used in wrongdoing examination, to gather those inert prints. Those two modalities were by a wide margin examined most and henceforth are considered mama ture in regards to acknowledgment execution and will keep on being used for the most part in minimal effort and multimodal frameworks.

Thusly the biometric look into group endeavored to discover new modalities that beat these downsides. One approach, that is just conceivable because of late mechanical advancements, is to infiltrate uninstructive into the human body and assemble data from that point. Covered up to the exposed eye and strong to the inactive duplicate issue, vein examples were found to be helpful in biometric verification.

Ordinarily, vein designs from the rear of the internal eye (retina acknowledgment) and vein designs from the appendages are recognized. The last is alluded to as vascular example acknowledgment or vein acknowledgment, the data begins for the most part from the hand range and is the concentration of this work. We recognize four sub-modalities: finger, palm, hand dorsal and wrist vein biometrics. We will likely enhance the acknowledgment execution of vein pat-terns from various hand-based modalities and above all to upgrade the security properties using protection improving advances (PETs) to conquer general issues of biometric frameworks. How such an upgraded biometric framework can be used as confirmation plan without the requirement for putting away touchy information is explored for one particular utilize case: the verification of web based managing an account exchanges.

2. Level of Research Already Carried Out on the Proposed Topic

No particular research has been carried out previously by the department in field of Dorsal Hand vein Biometrics. However, recent advances include M.Khalil Hani and P.C Eng [1] who introduced another procedure for person verification base on vein pattern on the back of the hand based on infrared, targeted for embedded system which is implemented in Altera Nios II FPGA prototyping system, running o-n Real Time Operating System (RTOS). Aycan Yuksel, Lale Akarun2 presented a novel biometric technique based on the statistical processing of the hand vein patterns. Xeu Yuan3 proposed a new grip-type hand view authentication system. Sang-Kyun Im, Hyung-Man Park, Soo-Won Kim, Chang-Kyung Chung* and Hwan-Soo Choi6 proposed an improved vein pattern extracting algorithm which compensates the loss of vein patterns in the edge area, gives more enhanced and stabilized vein pattern information, and shows better performance than the existing algorithm.

1.3 Reason/Justification for the Selection of the Topic

Keeping in mind the current security situation every organization and academic institutions need to be fully secure. The idea is to present the novel and efficient solution for this purpose. The project also has a wide scope in health sector for patient record. Attendance metering is one of the most used application. With the global economy and the rapid development of information technology, Biological feature recognition has become a research focus. Biological identification technology is more security and convenience over the traditional ones, and more attention has been paid to these areas. Vein recognition is a technology that lies on human vein image as its application basis. Human vein as an important identification feature has advantages such as uniqueness, stability, availability and non-touch. Fingerprint, on the other hand, is collected via touch, while face and voice vary with age and is subject to disguise.

1.4 Objectives

Our main objective is to develop contact free efficient identification hardware embedded system that provides acceptability, accuracy, speed, and robustness of technology. First Goal is to develop the system on Matlab via core generator and Simulink. With Reference models for database we will develop fast and efficient algorithms to get more enhanced and stabilized vein patterns with fast recognition speed and better FAR (False-acceptance Rate)4. Smoothing, edge-detection algorithms need to develop for feature extraction. After getting specific feature set image is set for verification phase where matching algorithms will detect the imposter or guanine candidate. Following is the proposed block diagram of the project.

1.5 Relevance to National Needs

Considering the security concerns in our country & old methods for identification & verification. It was a dire need to introduce a novel idea that will able to verify and authenticate individuals based on the irreplaceable, unmatched pattern. Thus, to fill this technological gap an application has implemented, cost-effective and which will identify the individual based on hand vein patterns. Such system can be used for patient's attendance in hospitals, verify the individuals in any organization and in many other industries.

1.6 Advantages

The importance of Biometric is significant both in terms of social and sociological perspective in terms of security & Surveillance, Attendance metering, Patient monitoring.

1.7 Areas of Application

With the global economy and the rapid development of information technology, Biological feature recognition has become a research focus. Areas of Application are wide as it ensues more secure nature. Biological identification technology is more security and convenience over the traditional ones, and more attention has been paid to these areas. Vein

recognition is a technology that lies on human vein image as its application basis. Human vein as an important identification feature has advantages such as uniqueness, stability, availability and non-touch

Chapter 2

Literary Review

The Demand of computational power and memory space has been fundamentally increment step by step. Remembering for the real time applications, image & video data have increasing gradually that requires more budget for storage purposes. E.g. cameras, projectors and so on. To cook such huge measure of information continuously applications with high throughput require some effective picture preparing calculations and usage. [1]. in current therapeutic and observation applications, picture handling and PC vision calculations broadly utilized because of high exactness and the overwhelming calculations. Constant frameworks should be precise, quick and ready to ascertain expansive calculations much speedier. The point of research is to keep up high precision and increment ongoing execution rate. Luckily, these calculations demonstrate a few parallelization openings, which help to quicken the handling when executed on multiprocessor stages. [2] To do ongoing execution a few issues need to analyze precisely, one is to build up an effective calculation and other one is to actualize it on speedier equipment stage. FPGA`s are noteworthy for this reason due to its parallel handling capacity and reconfigurable components, has turned into a helpful parallel stage for picture preparing. In [3], an equipment design in view of FPGA for picture obtaining and improvement i.e. histogram balance, differentiate extending and presentation control is proposed.

The research presents a novel idea to implement canny edge detection algorithm using CoreGen. System Generator for DSP™ is the industry's leading high-level tool for designing high-performance DSP systems using Xilinx All Programmable devices. With System Generator for DSP, create production-quality DSP algorithms in a fraction of time compared to traditional RTL [5]. The aim was to develop fast and efficient algorithm with low hardware resources which give fast recognition speed and provide better FAR (False-acceptance Rate). The system is implemented on System generator via MATLAB Simulink while the hardware platform used is Virtex-5 FPGA XtremeDSP™ system because of its capability and applicability.

It is a novel execution of a Canny edge finder that exploits 4-pixel parallel calculation. It is a pipelined engineering that utilizations on-chip BRAM recollections to reserve information between the distinctive stages. The misuse of both equipment parallelism and pipelining makes an extremely effective plan that has an indistinguishable memory necessities from an outline without parallelism in pixel calculation. This outcomes in accomplishing expanded throughputs for high determination pictures and a calculation time of 3.09ms for a 1.2Mpixel picture on a Spartan-6 FPGA. We exhibit amalgamation and recreation comes about for low-end and top of the line Xilinx FPGAs and have accomplished higher rates, preferable throughputs and effectiveness over the usage introduced previously.

2.1 Overview

This feature extracted from vein patterns that will verify the person after certain pre and post processing procedures. The algorithm consists of Image Acquisition, pre-processing of image, features extraction, and the verification module. The image processing, feature extraction as well as the authentication algorithm is performed using on FPGA using Hardware Co-Simulation technique via Core Generator. The hand vein extraction, which connection between points lying at the edges. Then matching algorithm used to identify the user as genuine & imposter candidate. 32

hand vein images from 16 different hands were used in this project. The minimum level or threshold is set to determine whether the image is Match or not.

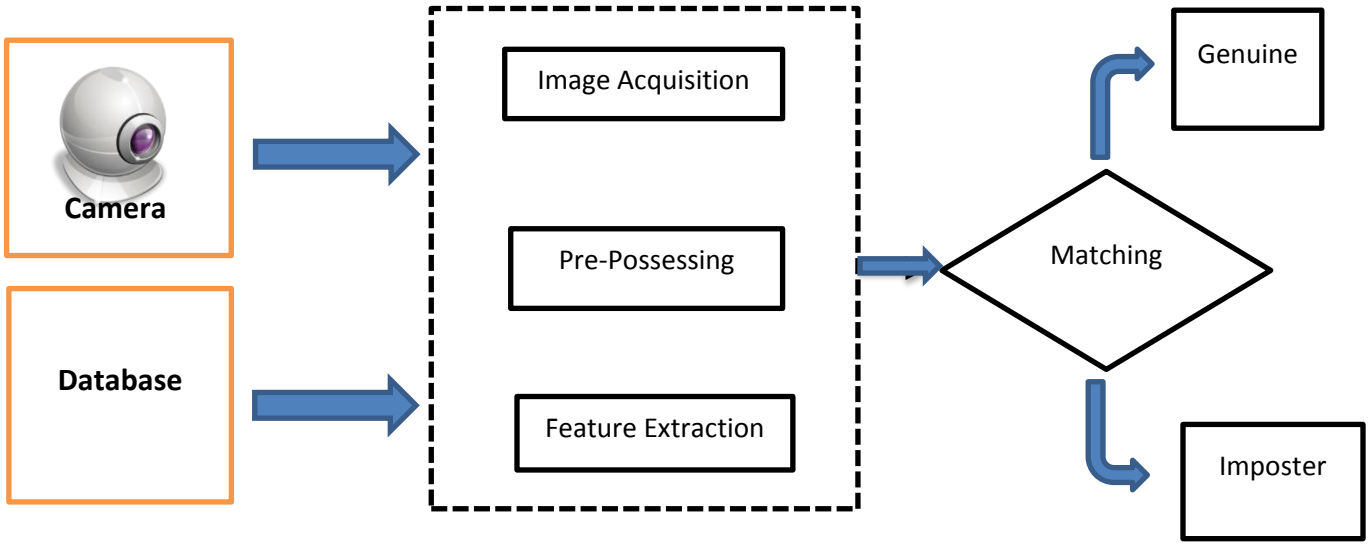

Fig1: Overall view of Proposed Research

2.2 Image Acquisition

In this work, we build a hand vein identification system with infrared camera to capture the image. The captured image then convert to Grayscale to transmit serially. Parameters were optimized according to the mean and variance of gray. Threshold method was used to remove the image background, and the image gray was normalized. It is show that the high quality images can be obtained by using Optimized parameters acquisition device and

image preprocessing procedure. The performance of vein recognition system is highly subject to the image quality; therefore the acquisition of high quality vein image is essential. But in practice, the acquisition of vein image comes under the influence of various possibilities such as palm thickness, vein size. These may lead to recognition errors by creating false and distorted features on images acquired by the system. Therefore, it is highly essential to automatically adjust the image acquisition system according to the evaluation of the acquired images, in order to obtain high-quality vein image. We have studied two types of spatial evaluation parameters.

Fig2: Hand Model from database image

2.3 Image Pre-Processing

Numerous executions of the Canny calculation discussed by various researchers. There is an arrangement of work [1]–[3] on Robert channels that was determined utilizing Canny's basis.

All the processing comprise of ASIC stages. The Robert channel [1] is a system that recognize edges in a 256 × 256 picture in 6s, a long way to use it for real time applications. In spite, the outline in [2] enhanced the Canny-Deriche channel usage of [1] and could prepare 25 outlines/s at 29 MHz, off-chip SRAM used on the basis on First-in Last out basis which expanded the range contrasted with [1]. Robert proposed another solution in [2] that was cost-effective and requires less memory. All the three cases required lesser clock cycles per pixel. Bringing about factor clock-cycles/pixel starting with one picture estimate then onto the next with expanding handling time as the picture measure increments.

Canny detector worsens quickly when managing salt and pepper commotion. And canny edge detection algorithm is computationally more edge recognition calculation is computationally more costly contrasted and others. Since, the span of the picture is duplicated now a days so it's getting to be distinctly hard to manage constant application. In this way, we build up a novel strategy to actualize shrewd edge identification calculation which is much productive for continuous applications. The framework is executed on Core Generator. The calculation created comprise of a few phases as appeared in the piece charts. The system is implemented on Core Generator. The algorithm developed consist of several stages as shown in the block diagram:

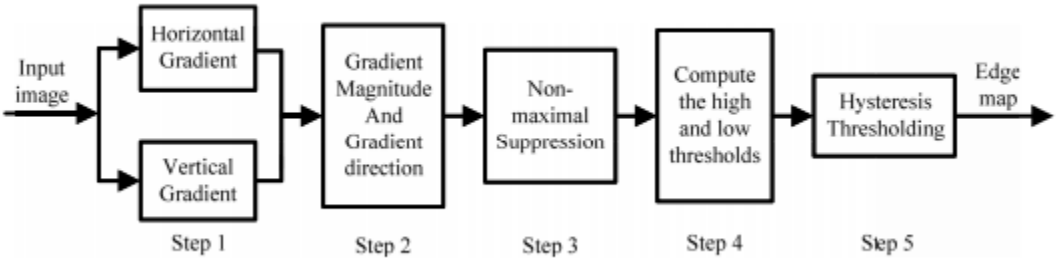

Fig3: Pre-possessing module

The stage consists of:

(1) Gaussian Blur, which lessens the commotion on the first orange by convolving the information picture with 2-D Gaussian Filter and Generates Horizontal and Vertical Gradients.

(2) Sobel Detection, which identifies the pixels power varieties in X and Y bearing by convolving the picture in every course of 2-D Gradient Filer.

(3) Magnitude, which ascertains the extent of every pixel.

(4) Direction, setting the heading of every pixel relying upon the pixel bearing of X and Y pixels.

(5) Non Maximum Suppression, which continues pixels with size more noteworthy than a low edge and spares pixels more prominent then a high edge in a rundown as solid edge pixels.

Chapter 3

FPGA IMPLEMENTATION OF THE PROPOSED SYSTEM

3.1 Architecture Overview:

Remembering the ultimate objective to define a viability of a proposed appropriated image identification calculation that portrays a FPGA-based gear use of the proposed figuring in this section which gives an elevated view point of view of structure of executing the pass**ed** on Filter count in perspective of a FPGA organize. It is made out of a couple sections, including an installed microcontroller, a structure transport, peripherals and periphery controllers. The engineering of the proposed conveyed Canny calculation. Outside Static memory and other controllers. An application has been designed for the proposed appropriated location calculation. The introduced littler scale controller encourages the trading of the photo from PC by Usb or PCI ports to static memories. Newer versions of Xilinx broads such as Virtex & Altera provide more libraries of core-generator as installed small scale ports. Accordingly, Main aim is to use Xilinx Altera for all massive calculations for all the convolution and other operations.

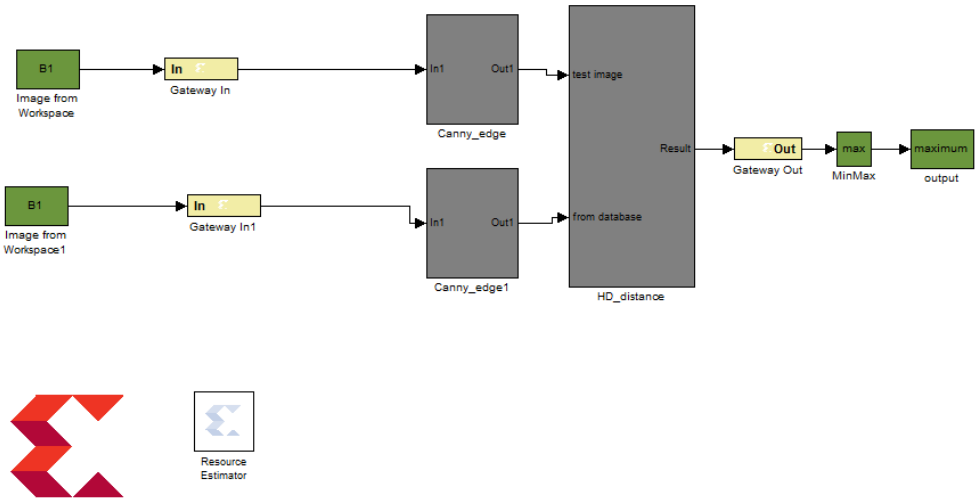

Fig4: Core-generator implementation blocks

3.2 Overall block Diagram:

As expressed, the n x n covering square is put away in the nearby m1 which was utilized for deciding the filter. Block Classification consists of 2 items: Filtration & Thresholding.

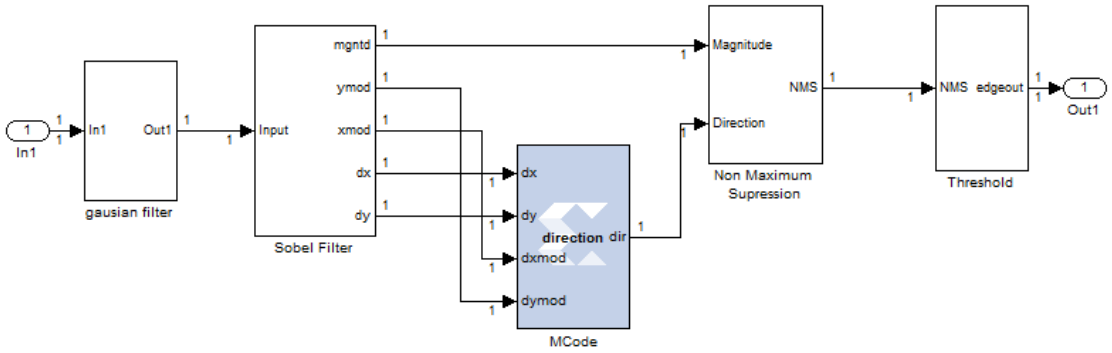

Fig5: Filtration (Pre-processing) block

The calculation is done utilizing one adder, 2 Multipliers, 2 Accumulators, and one square (move 3 bits to the right to accomplish increase by one-eight). Next, the neighborhood difference is contrasted and T_u and T_e [17] keeping in mind the end goal to decide pixel sort. At that point c1& c2 utilized to find the aggregate points for every pixel sort. The yield of counter c1 provides to counter c2. As soon as both counter accessible, all the segments with initialize. Counter 2 is compared to 0. The resulted outcome used as a flag of com5.

com6 and multiplex2 along with both counters are contrasted results in diverse values and the yields are utilized as signs of multiplexers which decide the estimation. At long last, the p1 esteem is contrasted with 0 with deliver the empower flag, set apart as EN. On the off chance that the P1 number is bigger than 0, then EN flag empowers angle count, greatness

computation, directional non-most extreme concealment, Fine and thin edge estimation and Hysteresis Threshold . Something else, all blocks don't should be initiated and the edge outline every one of the zero esteem pixels is put away once again into the SRAM. The p1 and en turn out to be yields for the piece order block that will put away in the register memory for Threshold measurements. The latency came out to be n x n+16 clock cycles between input image and out image of P1. The total execution time turned out to be n x n + 17

3.3 Sobel Edge-detection:

Both components i.e. Horizontal & Vertical Gradients will work in parallel. The n x n covering square is moreover secured in the CE's close-by memory 2 and 3 and is used as commitment to figure the even and vertical edge, independently. The designing for the slant and enormity figuring unit is showed up. Designing consists of Horizontal & vertical gradient; Threshold calculation).

Both Gradients calculation of the input image will convolve with 2d image, independently; the gradients can be find in perspective of the segments of the photo. These 2d incline cloak are particular, so by using two 1d convolution will be sufficient enough to get 2d convolution. To convolve the images we used FIR filters of core-generator, uses the same characteristics of the coefficient set. Subsequently, each convolve image is utilized with two Finite impulse response (FIR) Filters. Total 4 FIR filters will be used to get Horizontal & Vertical gradient. The outcome of FIR filter move to m2 and m3 respectively, independently, and are used as commitment to convolve image using columns one by one. Size calculation unit depends on the size output provided by FIR. The unit calculates the angle of both gradients. This Module includes mux1 mux2 and Sqr1. Square root unit is executed using CORDIC. The important factor for using core generator is pipelining. Truncated output stored in the neighborhood memory 1 for further count. The most outrageous and slightest estimations of the enormity, set apart as maxima and minima are the outputs of the unit.

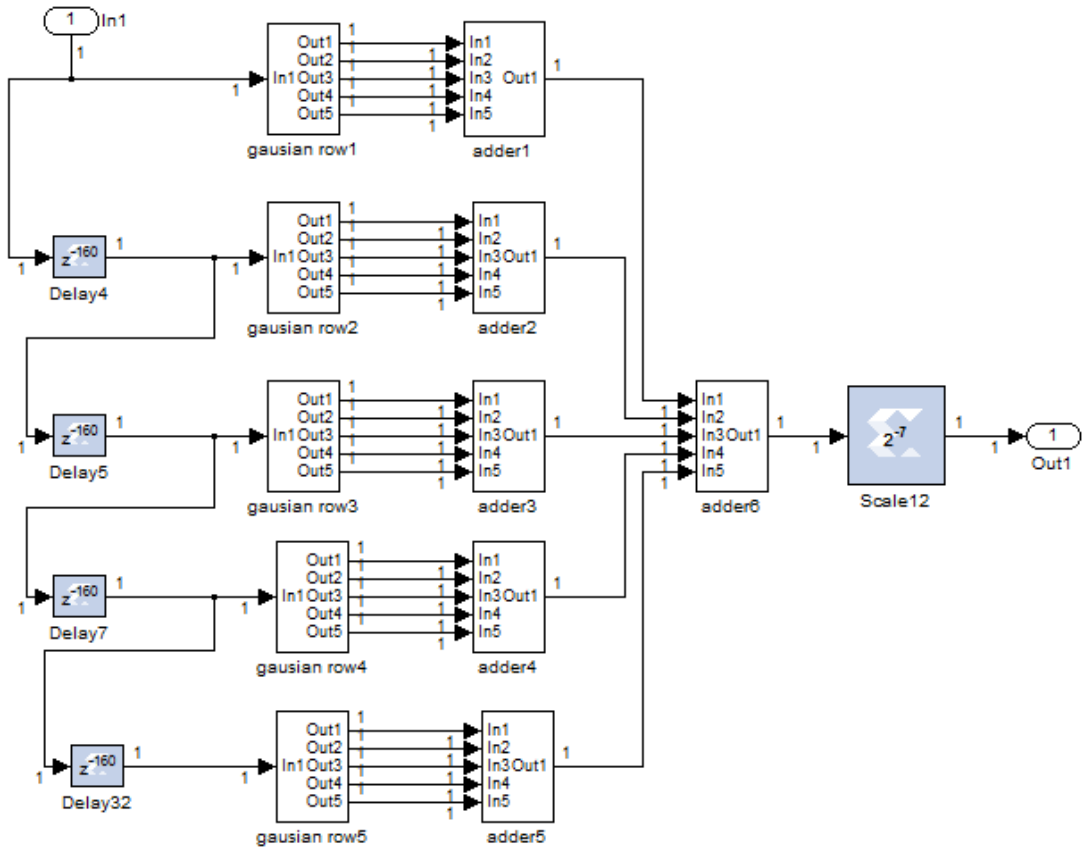

Fig6: Gaussian Filteration

3.4 Non-Maximum Suppression (NMS)

The Horizontal and vertical angles extent are brought from neighborhood i.e. m1, m2, m3 individually; and utilized as contribution to the non-maximum suppression. As depicted, this progression includes processing the slope bearing at every pixel. As indicated by the sign and estimation of the even slope Gx and the vertical inclination G y, the course can be

resolved; and in this manner the angle extents of four closest neighbors along the bearing are chosen to register two middle of the road inclinations M1 and M2. The level slope segment Gx and the vertical inclination segment Gy determine the selection which advances the size. Directional NMS most extreme concealment unit consists of (an) instrument; (b) design. The engineering of edges estimation unit. The heading of the slope into the math unit. This number juggling unit comprises of 1 divider, 1viper and 2 multipliers which are additionally actualized by the Math Functions of Xilinx. At last, the yield of the number juggling unit is contrasted and the inclination greatness of the middle pixel. The inclination of the pixel that does not relate to a neighborhood pixel will be consider as 0. As soon as NMS(x,y) along with angle found , the value passes to memory m1 which will used as a input to calculate threshold. Latency rate between Input image & NMS (x,y) is 20Clk Cycles and total time is n x n+ 25

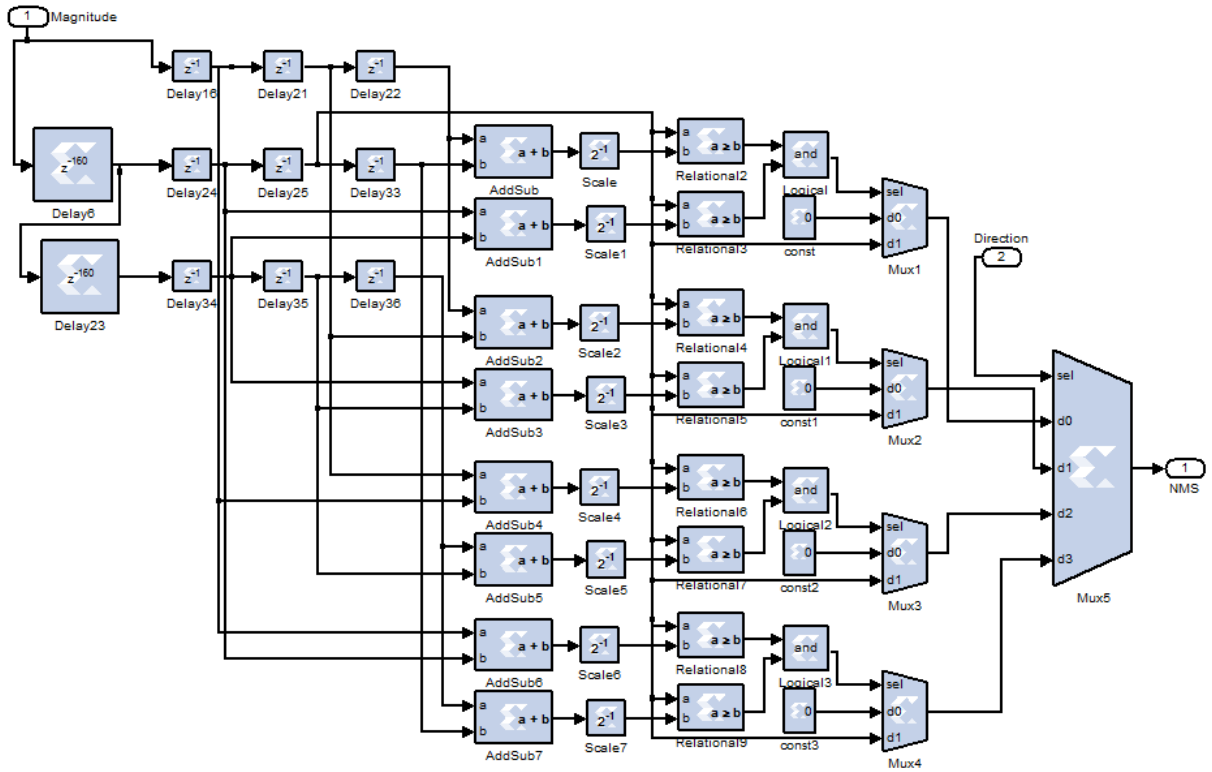

Fig7: Non-Maximum Separation (NMS) Module

3.5 Thresholds:

Magnitude of the gradient of every pixel will be use to find threshold. Pipelined building of thresholding with hysteresis. The module is pipelined with NMS also. The maxima & minima, and p1 are the inputs of this module. The 8-organize variable quantize is used to get the discrete random variable. These high and low thresholds calculation can be calculated based on the values of CDF. To obtain the reconstruction level adders & shifters used and to find the total size of corresponding numbers, all the counters and comparators used.

The P1 number multiplied by the total number of pixel values of single block results numpix_r_i. This result then compared with numpix_ri to obtained level i. At this level, no pixel of r_i is close to p_i

3.6 Hysteresis Threshold

The inclination greatness of every point after non-maximum suppression is gotten from neighborhood memory 1 and utilized as contribution to the Threshold unit. In the interim, the T_h and T_{hl}, which are dictated by the edge computation block, are additionally the contributions for this unit. F1 speaks to a solid edge pixel while f2 speaks to a powerless edge pixel. If any pixel in neighbor is a solid edge pixel then the central lesser edge pixel will be consider as strong edge. It is considered as a foundation edgeless point. Latency find to be 10 cycles between first inputs. Total execution time turned out to be m x m + 15 cycles.

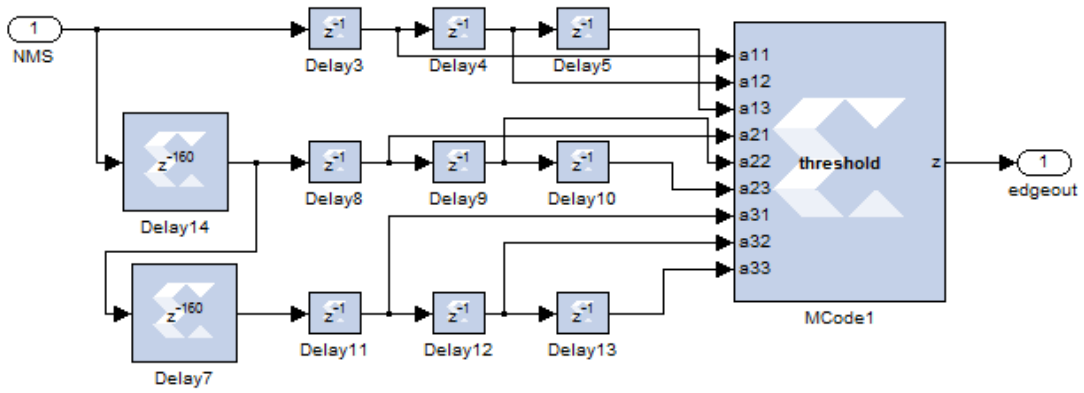

Fig8: Calculating Hysteresis Threshold

3.7 Matching

Strategies for middle channel on FPGAs have as of now been explored. Sobel operation has a place with the class of straight move invariant (LSI) operations. It is appropriate for a FPGA execution. Along these lines, the pipelined preprocessing can yield one component of the picture in each clock cycle. For the separation change, a FPGA usage is proposed in this paper, which likewise creates an outcome in each clock cycle. We will portray it in detail in the following subsection. Sadly, the format coordinating subtask takes a great deal additional time than the others. While contrasting a layout picture and a bigger target picture, we more often than not require that the coordinating calculation doles out an incentive for each conceivable overlay position of the format picture over the objective picture, which measures how great the match is. For every overlay position, the count of the coordinating measure, incomplete Hausdorff remove, needs at most Ht*Wt cycles (Ht and Wt signifies the stature and width of the format separately). Along these lines, the format coordinating is the most tedious stage in the framework

3.7.1. FPGA Implementation of Distance Transform*:*

As portrayed in Section II-B, the chamfer 3-4 separate change is received in this paper. A non-symmetric forward and in reverse cover is utilized as delineated in Fig. 7. The DT yield of current pixel is the base of the weighted pixel values under the cover. Since the forward and in reverse DTs are comparative, we concentrate on the forward DT.

Despite the fact that these area changes are all the time ascertained by moving the cover line by line over the picture, we actualize the DT the other path round: the veil is hardwired into the FPGA and the picture is interpreted under the cover. The pipeline of the DT is outlined in Fig. 8. Current pixel is contrasted and the base of the three comparing pixels in the previous line. The transitional outcome must be contrasted and the neighboring pixel which is now put away in an enlist. The outcome is composed to the on-chip memory, the off-chip memory (work pool) and is likewise put away in an enlist subsequent to being included by 3. This pipeline is equipped for creating another outcome in each clock cycle. The asset use of the separation change for 8-bit pictures with size of 1024x1024 is given in Table I. A higher clock speed can be accomplished subsequent to advancing the place and course (P&R) of the plan.

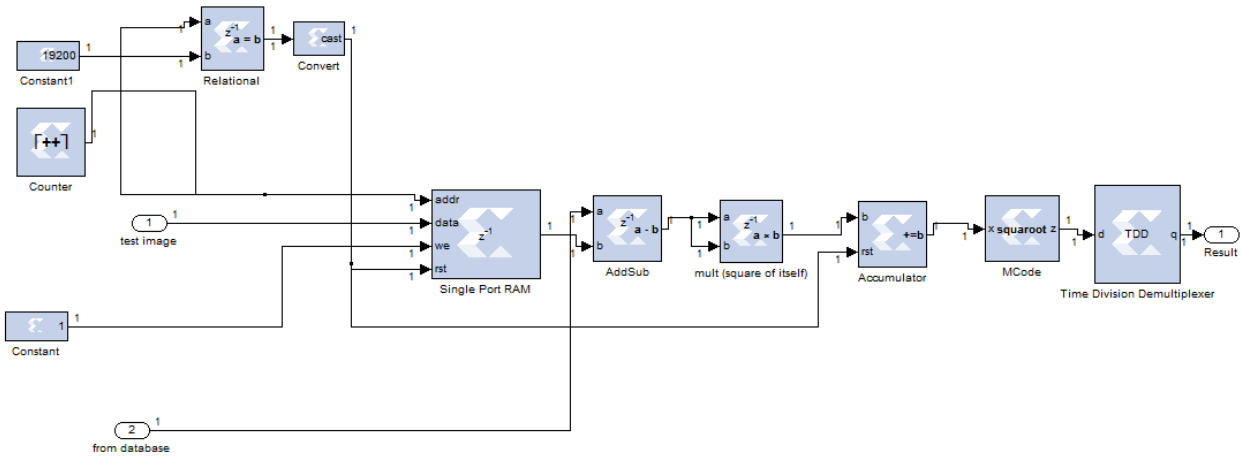

Fig9: Non-Maximum Separation (NMS) Module

The following after stride is to check for which personality the test class has a place with. As it were, the test picture is checked for which known class it is near. Truth be told, every vein picture has a place with a class with their weights. Euclidean separation is utilized as a measure of order between the test class and all the current classes in the database. In this way, utilizing this weight vector, we compute the Euclidean separation as takes after:

$$\in_k^2 = \| \boldsymbol{\Omega_p} - \boldsymbol{\Omega_k} \|^2$$

Where $1 \leq k \leq 1$, Ω_p is the is the weight vector relating to the test vein picture, that is,

$\Omega_p = [w_{p1}, \ldots\ldots w_{pm}]$ and Ω_k is the weight vector corresponding to the vein image of the k^{th} individual (kth vein class), that is $\Omega_k = [w_{k1}, \ldots\ldots w_{km}]$. The separation acquired is checked against an edge esteem, θ_{class}. This Threshold esteem was acquired from all tests completed with a dorsal hand vein framework created.

3.7.2 Authentic Score and Imposter Score

Authentic Score is the Euclidean separation between the testing set and the standard arrangements of the considerable number of people. Then again, imposter score is the Euclidean separation between the testing set and the arrangement of every other individual in the layout set (less the cases of a similar individual). The arrangement of certifiable scores and the imposter scores rely on upon the quantity of test veins and the quantity of format veins. In this work, pictures were gotten from 100 people of various skin hues, race and ethnic gatherings were caught. Ten occurrences of dorsal hand vein example were caught for every person out of which 5 were kept for the format set and 5 were kept for the testing set. The aggregate check of real scores is 100x5x5=2500. Fraud scores are likewise produced and the aggregate tally of the faker score is 500x475=237500. As clarified to a limited extent An in this area, an edge esteem is processed in light of the authentic score and sham score and is itemized in the accompanying segment.

Experimental Results

The Genuine score circulation and the imposter score conveyance is appeared in the figure 1 and 2 separately:

At certain point the Authentic score and the imposter score will cover. Accordingly, an ideal limit must be picked. On the off chance that the match score is not as much as the picked esteem, the test is perceived as a bona fide client, else, on the off chance that it is more noteworthy, the test is perceived as a sham. To test the framework, false acknowledgment rate (FAR), which is the quantity of times a fraud is considered as a genuine client and false dismissal rate (FRR), which is the quantity of times a genuine client is considered as a faker, have been processed.

The trial was completed utilizing distinctive vein design for the format and for the test to be more reliable. The normal is then ascertained as appeared in the table beneath:

Threshold	**False Acceptance Rate**	**Rejection Rate**
0.68	3.1	2.8
0.61	2.9	2.1
0.71	1.49	1.31
0.96	0.59	0.01
1	1.19	0.98

Table01: Matching Results

Fig11: Input & Output image of hand

Conclusion

The method can support diverse pictures of any size. In order to define the execution method of this particular research. Virtex-7 FPGA Family is used to deal with b&w pictures with a square size of 32 × 32. Thus the width to be of 16-bits, image value is 255, and fragmentary part used 7 bits cause of decimal parameters of Gaussian filter. Our examination shows that Testing shows that to reach at the level of precision 7 bits are enough. The principal Canny computation relies on upon edge level estimations to foresee the high and low breaking points and in this way has torpidity comparing to the packaging size. Remembering the ultimate objective to lessen the immense inertness and meet continuous necessities, we showed a novel passed on Canny edge area estimation which can enlist edges of various pieces meanwhile.

To support this, an adaptable point of confinement assurance procedure is introduced that determine the high and low points of the entire image by simply taking care of the points of every block. Outcome consists of 3 points of interest:

- Reduce latency rate.
- Enhanced edge-deduction.

Pipelining with all other blocks.

Besides, less versatile quality variable normalized histogram figuring procedure computes the block`s minimum level. Defined method is measurable, versatile & robust. It is highlighted that our method can detect all the edges and boundaries of any image of all sizes. In the end, the computation is built on Virtex-7 FPGA Family along-with Core-Generator/ModelSim. The robust outcome shows 65% slices and 83% memory usage of BRAM. Implementation took around 0.7ms to recognize edges of 32 × 32 pictures at 100 MHz. Thus, proposed research is fast, accurate and more robust as compared to previously

used techniques. This method can detect edges of all real time images and video applications.

References

[1] R. Deriche, "Using canny criteria to derive a recursively implemented optimal edge detector," Int. J. Comput. Vis., vol. 1, no. 2, pp. 167–187, 1987.

[2] L. Torres, M. Robert, E. Bourennane, and M. Paindavoine, "Implementation of a recursive real time edge detector using retiming technique Proc. Asia South Pacific IFIP Int. Conf. Very Large Scale Integr., 1995, pp. 811–816.

[3] F. G. Lorca, L. Kessal, and D. Demigny, "Efficient ASIC and FPGA implementation of IIR filters for real time edge detection," in Proc.IEEE ICIP, vol. 2. Oct. 1997, pp. 406–409.

[4] D. V. Rao and M. Venkatesan, "An efficient reconfigurable architecture and implementation of edge detection algorithm using handle-C," in Proc. IEEE Conf. ITCC, vol. 2. Apr. 2004, pp. 843–847.

[5] H. Neoh and A. Hazanchuck, "Adaptive edge detection for real-time video processing using FPGAs," Altera Corp., San Jose, CA, USA, Application Note, 2005.

[6] C. Gentsos, C. Sotiropoulou, S. Nikolaidis, and N. Vassiliadis, "Realtime canny edge detection parallel implementation for FPGAs," in Proc. IEEE ICECS, Dec. 2010, pp. 499–502.

[7] W. He and K. Yuan, "An improved canny edge detector and its realization on FPGA," in Proc. IEEE 7th WCICA, Jun. 2008, pp. 6561–6564.

[8] I. K. Park, N. Singhal, M. H. Lee, S. Cho, and C. W. Kim, "Design and performance evaluation of image processing algorithms on GPUs," IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 1, pp. 91–104, Jan. 2011.

[9] J. D. Owens et al., "A survey of general-purpose computation on graphics hardware," Comput. Graph. Forum, vol. 26, no. 1, pp. 80–113, 2007.

[10] Y. Luo and R. Duraiswami, "Canny edge detection on NVIDIA CUDA," in Proc. IEEE CVPRW, Jun. 2008, pp. 1–8.

[11] R. Palomar, J. M. Palomares, J. M. Castillo, J. Olivares, and J. Gómez-Luna, "Parallelizing and optimizing lip-canny using NVIDIA CUDA," in Proc. IEA/AIE, Berlin, Germany, 2010, pp. 389–398.

[12] L. H. A. Lourenco, “Efficient implementation of canny edge detection filter for ITK using CUDA,” in Proc. 13th Symp. Comput. Syst., 2012, pp. 33–40.

[13] Q. Xu, C. Chakrabarti, and L. J. Karam, “A distributed Canny edge detector and its implementation on FPGA,” in Proc. DSP/SPE), Jan. 2011, pp. 500–505.

[14] J. F. Canny, “A computation approach to edge detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 8, no. 6, pp. 769–798, Nov. 1986.

[15] S. Nercessian, “A new class of edge detection algorithms with performance measure,” M.S. thesis, Dept. Electr. Eng., Tufts Univ., Medford, MA, USA, May 2009.

[16] P. Bao, L. Zhang, and X. Wu, “Canny edge detection enhancement by scale multiplication,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 9, pp. 1485–1490, Sep. 2005.

[17] J. K. Su and R. M. Mersereau, “Post-processing for artifact reduction in JPEG-compressed images,” in Proc. IEEE ICASSP, vol. 3. May 1995, pp. 2363–2366.

[18] P. Arbelaez, C. Fowlkes, and D. Martin. (2013). The Berkeley Segmentation Dataset and Benchmark [Online]. Available: http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/

[19] N. D. Narvekar and L. J. Karam, “A no-reference image blur metric based on the cumulative probability of blur detection (CPBD),” IEEE Trans. Image Process., vol. 20, no. 9, pp. 2678–2683, Sep. 2011.

[20] (2013). USC SIPI Image Database [Online]. Available: http://sipi.usc.edu/database/database.php?volume=misc

[21] (2013). Standard Test Image Database [Online]. Available: http://www.imageprocessingplace.com/

[22] L. V. Scharff, A. Hill, and A. J. Ahumada, “Discriminability measures for predicting readability of text on textured backgrounds,” Opt. Exp., vol. 6, no. 4, pp. 81–91, Feb. 2000.

[23] (2013). Altera Available IP [Online]. Available: http://www.altera.com/products/ip/processors/nios2/features/ni2-peripherals.html

[24] (2013). Xilinx Available IP [Online]. Available: http://www.xilinx.com/ise/embedded/edk/_ip.htm

[25] (2007). Xilinx IP LogiCore FIR Datasheet [Online]. Available: http://www.xilinx.com/support/documentation/ip_documentation/fir_compiler_ds534.pdf

[26] (2012). Xilinx Vertrix-5 Family Datasheet [Online]. Available:

http://www.xilinx.com/support/documentation/data_sheets/ds100.pdf

Appendix A:

==========Code for Edge Direction==========

```
function dir = direction(dx, dy, dxmod, dymod);
if( dx>=0 & dy >= 0)
    if(dxmod >= dymod)
        dir = 0;
    else
        dir = 1;
    end
elseif (dx<=0 & dy>=0)
    if(dxmod <= dymod)
        dir = 2;
    else
        dir = 3;
    end
elseif (dx<=0 & dy<=0)
    if(dxmod >= dymod)
        dir = 0;
    else
        dir = 1;
    end
else
    if(dxmod <= dymod)
        dir = 2;
    else
        dir = 3;
    end

end


end
```

==========Code for Counters==========

```
function rst=conters(x)

max=19201;

if (x==max)

   rst=1;

else
```

```
    rst=0;
end
```

==========Code for Delete User==========

```
function []=deleteuser(name,course)
load('C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\database.mat');
i=1;
while i<length(s)
 if  ((strcmpi( name,s(i).name)==1) && (strcmpi(course, s(i).course )==1))
  h = msgbox('User Deleted',s(i).name )
      j=i+1;
      s(1,i:end-1)=s(1,j:end);
        s(1,end).name=[];
       s(1,end).index=[];
       s(1,end).course=[];

    save('C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\database.mat','s')
      break
    else
     i=i+1;
  end
end
i;
if (i==length(s))
   errordlg('User not found. Try agin','User doesnt exist');
end
```

==========Code for Image Transfer to Database==========

```
s=imread('C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\hand database\38.jpg');
A=imresize(s,.25);
B2=rgb2gray(A);
imshow(B2);
B1=im2serial(B2);
```

==========Code for Converting Data to Serial==========

```
function [ser] = im2serial(d)
[a b] = size(d);
ser = d(1, :);
for i= 1 : a-1
   ser = [ser d(1+i,:)];
end
```

==========Code for Loading Data to ROM==========

```
function [text2]=load_data()
load('C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\database.mat')
n=10;%how many images you want to write
for i=1:n
t{i}=s(i).image;
end
d=cell2mat(t');
out=im2serial(d);
[c nbits]=size(out);
out=out';
nbits1=nextpow2(nbits);
```

```
% disp('No of bits required for addressing the Memory');disp(nbits1);

% disp('Depth of  Memory');disp(nbits);

% text1='loading data into ROM.....';

%disp('loading data into ROM.....');

dlmwrite('C:\Users\TalhaAbdulQayyum\Desktop\Biometric\vernew1\hand
database\data.txt',out','newline', 'pc');

%disp('Completed');

text2='Completed';
```

==========Code for Manualy Adding data to ROM==========

```
% function []=add_data()

TargetFolder = 'C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\';

myFolder = 'C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\hand';


if ~isdir(myFolder)

  errorMessage = sprintf('Error: The following folder does not exist:\n%s', myFolder);

  uiwait(warndlg(errorMessage));

  return;

end

filePattern = fullfile(myFolder, '*.jpg');

jpegFiles = dir(filePattern);

jpgdata = cell(1,length(jpegFiles));

% for k=1:length(jpegFiles)

for k=1:10

   prompt = {'Enter your name:','Enter your course:'};

   dlg_title = 'Input for new entry database';

   num_lines = 1;
```

```
    def = {'name','course'};
    answer = inputdlg(prompt,dlg_title,num_lines,def);
    n=answer(1);
    j=answer(2);
    close all
    baseFileName = jpegFiles(k).name;
    disp(baseFileName);
    fullFileName = fullfile(myFolder, baseFileName);
    jpgdata{k}= imread(fullFileName);
    rescale_data{k}=imresize(jpgdata{k},0.25);
    conver_clr{k}=rgb2gray(rescale_data{k});

    s(k)=struct('index',[k],'name',n,'course',j,'image',conver_clr{k});
end
namedir='database';
targetdir=fullfile(TargetFolder, namedir);
save(targetdir,'s')




% data_addition(TargetFolder);
% prompt = {'Enter your name:','Enter your course:'};
% dlg_title = 'Input for new entry database';
% num_lines = 1;
```

```
% def = {'name','course'};
% answer = inputdlg(prompt,dlg_title,num_lines,def);
% n=answer(1);
% j=answer(2);
% % load('D:\database4.mat');
% % v=length(s);
%
% % x1=image;
% % imwrite(x1,['C:\Documents and Settings\ammar\My Documents\My Pictures\',num2str(v+1),'.jpg']);
```

==========Code for Matching two Images==========

```
function varargout = matching(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @matching_OpeningFcn, ...
                   'gui_OutputFcn',  @matching_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end


if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
function matching_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to matching (see VARARGIN)

% Choose default command line output for matching
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes matching wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% axes(handles.LoadedImage)
% imshow('05.jpg')
% axes(handles.MatchingImage)
% imshow('06.jpg')
global vid

[a,map]=imread('C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\blue.png');
[r,c,d]=size(a);
```

```
x=ceil(r/57);
y=ceil(c/200);
g=a(1:x:end,1:y:end,:);
g(g==255)=5.5*255;
set(handles.MinutaePointMatch,'CData',g);

[a,map]=imread('C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\blue.png');
[r,c,d]=size(a);
x=ceil(r/57);
y=ceil(c/200);
g=a(1:x:end,1:y:end,:);
g(g==255)=5.5*255;
set(handles.LoadImage,'CData',g);
[a,map]=imread('C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\blue.png');
[r,c,d]=size(a);
x=ceil(r/57);
y=ceil(c/237);
g=a(1:x:end,1:y:end,:);
g(g==255)=5.5*255;
set(handles.HammingDistMatch,'CData',g);

function varargout = matching_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function LoadImage_Callback(hObject, eventdata, handles)
function pushbutton2_Callback(hObject, eventdata, handles)
```

```
function MinutaePointMatch_Callback(hObject, eventdata, handles)
maximum = evalin('base', 'maximum')
axes(handles.axes5);
maxdisp(maximum);
guidata(hObject,handles)
% --- Executes on button press in UserMatch.
function HammingDistMatch_Callback(hObject, eventdata, handles)
% hObject    handle to HammingDistMatch (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%IdentifyGUI=getappdata(0,'IdentifyGUI');
%cropimg=getappdata(IdentifyGUI,'crpimg');
% % IdentifyGUI=getappdata(0,'IdentifyGUI');
% % cropimg=getappdata(IdentifyGUI,'crpimg');
% % axes(handles.MatchingImage);
% % HammingMtch(cropimg)
%%handles.local_maximum = maximum;
% IdentifyGUI=getappdata(0,'IdentifyGUI');
% name=getappdata(IdentifyGUI,'name');
% course=getappdata(IdentifyGUI,'course');
% axes(handles.MatchingImage);
%check_data(maximum,10)

maximum = evalin('base', 'maximum')
% dataimages= evalin('base', 'dataimages')
```

```
[name1 course1 image1 status]=check_data(maximum)
caption = sprintf('Name: %s',name1)
set(handles.ST2, 'String',caption);
cap = sprintf('Course:%s',course1)
set(handles.ST1, 'String',cap);

axes(handles.axes9);

imshow(image1)
axis off;
 guidata(hObject,handles)
 ca = sprintf('Status:%s',status)
set(handles.ST4, 'String',ca);
%setappdata(IdentifyGUI,'minutaepoints',minutaepnts);


% --- Executes during object creation, after setting all properties.
function LoadedImage_CreateFcn(hObject, eventdata, handles)
% hObject    handle to LoadedImage (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
function MatchingImage_CreateFcn(hObject, eventdata, handles)
% hObject    handle to MatchingImage (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: place code in OpeningFcn to populate MatchingImage
% --- Executes during object creation, after setting all properties.
function axes4_CreateFcn(hObject, eventdata, handles)
axes(hObject)
imshow('C:\Users\Talha Abdul Qayyum\Desktop\Biometric\vernew1\greenback.png')
% --- Executes on selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns listbox1 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from listbox1
% --- Executes during object creation, after setting all properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
function ST1_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to ST1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of ST1 as text
%        str2double(get(hObject,'String')) returns contents of ST1 as a double
% --- Executes during object creation, after setting all properties.
function ST1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ST1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
```

==========Code for Calculating Maximum Distance==========

```
function []=maxdisp(maximum)
dataimages=1;  % number of database images
dataimages=dataimages+1; % 1 for delayed zero
[value,index]=min(maximum([2:dataimages],1));

if (value<=40)

   disp('image matched.....');
   disp('minimum hausdoof distance is =' );disp(value);
```

```
   disp('index=');disp(index);


   y=maximum([2:dataimages],1);

   x=1:dataimages-1;

   bar(x,y,0.00003,'r');

   title({['Image number # ',int2str(index),' is matched'] ;['minimum hausdoof distance is = ', int2str(value)]});

else


   disp ('image not matched.... try again dude... ');


end
```

==========Code for Calculating Square Root==========

```
function z = squaroot(x)

max = x;

min=0;

% for i=1:128

%  mid=(max+min)/2;

%  mid_2=mid*mid;

%  if (mid_2 > x)

%     max=mid;

%     z = mid;

%  else

%     min=mid;

%     z = mid;

%  end
```

```
  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end
  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
```

```
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
```

```
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end
  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
```

```
  min=mid;
end
  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end
```

```
  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end
```

```
  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
```

```
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
if (mid_2 > x)
  max=mid;
else
  min=mid;
end

  mid=(max+min)/2;
  mid_2=mid*mid;
```

```
if (mid_2 > x)
   max=mid;
else
   min=mid;
end

   mid=(max+min)/2;
   mid_2=mid*mid;
if (mid_2 > x)
   max=mid;
   z = mid;
else
   min=mid;
   z = mid;
end
end

function ST2_Callback(hObject, eventdata, handles)
% hObject    handle to ST2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ST2 as text
%        str2double(get(hObject,'String')) returns contents of ST2 as a double
```

```
% --- Executes during object creation, after setting all properties.
function ST2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ST2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function ST4_Callback(hObject, eventdata, handles)
% hObject    handle to ST4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ST4 as text
%        str2double(get(hObject,'String')) returns contents of ST4 as a double


% --- Executes during object creation, after setting all properties.
```

```
function ST4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ST4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
```