Efficient Fingerprint Image Enhancement for Mobile Embedded Systems

J. S. Chen, Y. S. Moon, K. F. Fong

Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T. Hong Kong
{jschen, ysmoon, kffong}@cse.cuhk.edu.hk

Abstract. Fingerprint image enhancement is an important step in a fingerprint verification system. The enhancement process, however, are often not applied to mobile embedded devices in which floating-point processing units (FPU) are absent. Earlier Hong and Jain reported a fingerprint enhancement algorithm based on the Gabor Filter. This algorithm and its derivatives have been proved to be quite effective in improving the fingerprint verification reliability. In this paper, we present an efficient implementation of this algorithm in an embedded system environment. In our implementation, fixed-point arithmetic is used to replace the floating-point operations. Moreover, a special Gabor filter parameter selection constraint is also proposed to reduce the computing complexity of the kernel generation step. Experimental results show that our new approach achieves significant speed improvement and is almost as effective as the traditional floating-point based implementation.

1 Introduction

Fingerprint verification systems are now widely used for commercial and security purposes since they are convenient, cheap and relatively superior to other biometrics systems. Recent development in hardware has made it possible to incorporate fingerprint verification systems in such embedded environments as door access, PDA or even mobile phone authentication applications. In these embedded fingerprint verification systems, the important step of fingerprint image enhancement is always skipped because most embedded devices are equipped with RISC processors like the ARM and StrongARM processors without hardware floating-point processing units (FPU) while fingerprint image enhancement procedures usually require substantial quantities of floating-point operations. Such omissions can cause downgrades of the performances of most fingerprint verification systems when handling low quality fingerprint images due to predictable factors such as thin ridges, scars as well as unpredictable factors such as dry/wet fingers, moving fingers. In view of such a difficulty, creating an efficient fingerprint image enhancement implementation for the embedded environments becomes an urgent task.

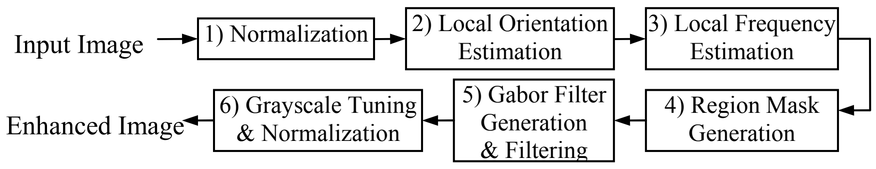
Among the many proposed approaches for fingerprint image enhancement [3], [4], [5], [6], the Gabor filter based algorithm reported by Hong et al. [3] has been commonly regarded as an effective mean for improving the reliability of the fingerprint verification process [10], [12]. As the algorithm aims at the online fingerprint verification system, its computational efficiency is quite suitable for the applications in which speed is critical. Therefore, our implementation is based on this algorithm. The speed optimization starts with the replacement of floating-point operations by fixed-point operations similar to work reported in [1], [8]. We then propose a simple Gabor filter parameter selection scheme that speeds up the Gabor filter kernel generation.

The rest of this paper is organized as follows. Section 2 introduces the outline of the enhancement algorithm as well as the structure of the complete system. Section 3 gives the detailed descriptions of the implementation. In section 4, we analyze the influence of the SVD on the convolution process. Experiment procedures and results are elaborated in section 5. The last section is a conclusion of our work.

2 Previous Work

2.1 Fingerprint Enhancement Algorithm

Fig. 1 shows the major steps of the fingerprint image enhancement algorithm [3]. This algorithm treats a gray-scale fingerprint image as a flow-like patter consisting of ridges and valleys. It assumes that the ridges and valleys form a sinusoidal-shaped wave with well-defined frequency and orientation varying smoothly and slowly along the flow of the ridges and valleys in a local area. After normalization in which the gray scale variations along the ridges and valleys are smoothed, the fingerprint image is divided into blocks of a suitable size. Local ridge orientation and frequency are estimated for each image block. Then, a Gabor filter kernel tuned to the estimated local orientation and frequency is applied to each image block to generate the enhanced image.



Input Image
1) Normalization
2) Local Orientation Estimation
3) Local Frequency Estimation
4) Region Mask Generation
5) Gabor Filter Generation & Filtering
6) Grayscale Tuning & Normalization
Enhanced Image

Fig. 1. The fingerprint image enhancement process

To adapt the image enhancement algorithm to our direct gray-scale based fingerprint features extraction system [7], an extra step, step 6, is inserted to tune the gray scale levels in the enhanced fingerprint images. To ensure the gray-scale consistency of the enhanced images, local normalization is applied to each image block after the gray-scale tuning.

2.2 System Structure

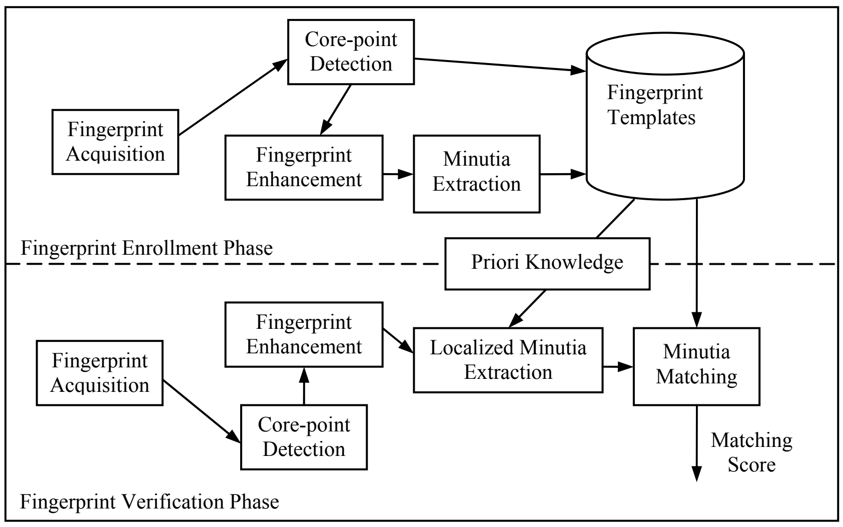


Fig. 2. The structure of the embedded fingerprint verification system with fingerprint image enhancement step

After adding the fingerprint image enhancement, the system structure [2] of the embedded fingerprint verification system is shown in Fig. 2. We can observe that fingerprint image enhancement is performed before the minutia extraction process both in the fingerprint enrollment phase and the fingerprint verification phase.

3 Implementation

To realize fingerprint image enhancement in mobile embedded systems with time efficiency, it is important that the mathematical approximations and constraints that we use to optimize the computational time do not affect the effectiveness of the image enhancement as well as the reliability of the ultimate fingerprint verification results significantly.

3.1 Fixed-point Arithmetic

The use of fixed-point arithmetic in the embedded biometrics systems has been studied in [1], [8], [18]. In our fingerprint image enhancement implementation, we have adopted a similar approach to facilitate the whole process in the embedded environment. A commonly used representation of a fixed-point arithmetic system [9] is shown in Fig. 3.

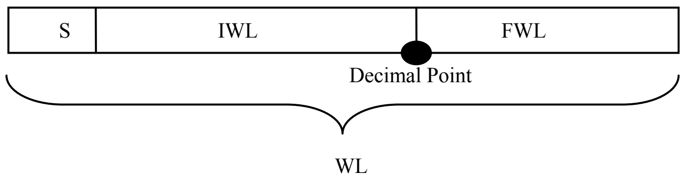


Fig. 3. Representation of the fixed-point arithmetic system: S-unsigned or two's complement (signed); WL-word length; IWL-integral word length; FWL-fractional word length

Suppose $I(i,j)$ and $N(i,j)$ are the gray-level values of the i-j pixel in the input image and the normalized image respectively. The normalization process can be defined as

$$N(i,j) = round\left(M_0 + \frac{STD_0}{STD} \times \left(I(i,j) - M \right) \right) \tag{1}$$

in which M and STD denote the mean and standard deviation of I, M_0 and STD_0 are the desired mean and standard deviation. Calculation of M and STD can involve large intermediate values due to the summation of the corresponding values from all the pixels. We, therefore, choose to use integer (32 bit) operation for this step. Since values of M_0, STD_0, M, STD and $I(i,j)$ are smaller than 256, selecting IWL=8 is enough for their numerical ranges. As $|I(i,j) - M|$ is always smaller than 256, if the numerical precision of STD_0/STD is smaller than 1/256, the maximum error of the final normalization result is 1 gray-level. This leads to FWL=8. Similar analyses are performed to the other enhancement steps. Table 1 shows the results of accuracy analyzing for each enhancement step in our implementation.

Table 1. Accuracy analysis results for each enhancement step

Enhancement Step	**Fixed-point Arithmetic Representation**	**Maximum Error**
Normalization	8.8	1 Gray Level
Local Orientation Estimation	8.6	0.006
Local Frequency Estimation	13.10	0.001
Region Mask Generation	--	--
Filtering	15.8	1.2%
Local Gray-scale Tuning & Local Normalization	8.8	1 Gray Level

Obviously, by using the 15.16 representation, the computational error of our implementation is not bigger than the values listed in Table 1.

3.2 Gabor Kernel Generation

Gabor filters have the properties of supporting both frequency-selection and orientation-selection [16]. It is appropriate to apply Gabor filters to the fingerprint image to remove the noises and sharpen the ridges and valleys [3], as the local area of a fingerprint is quite close to a sinusoidal shaped wave consisting of ridges and valley. As proposed in [3], [11], the even-symmetric Gabor filter used in the fingerprint image enhancement has the general form of

$$g(x, y; \phi, f) = \exp\left\{-\frac{1}{2}\left[\frac{x_\phi^2}{\delta_x^2} + \frac{y_\phi^2}{\delta_y^2}\right]\right\} \times \cos(2\pi f x_\phi) \tag{2}$$

$$x_\phi = x\cos\phi + y\sin\phi \tag{3}$$

$$y_\phi = -x\sin\phi + y\cos\phi \tag{4}$$

where ϕ is the orientation of the Gabor filter, f is the frequency of the sinusoidal wave, δ_x and δ_y are the standard deviations of the Gaussian envelops along the x and y axes, respectively. If the size of the Gabor filter is $2w+1$, the values of x and y should be integers within the range of $[-w, +w]$.

Generating the Gabor kernel during fingerprint image enhancement is a time consuming process since one Gabor filter has to be generated for every image block. From (2), (3) and (4), we can see that the generation process involves add, multiple, trigonometric operations and exponential operations. To speed up the trigonometric operations, table lookup techniques have been used in the fixed-point arithmetic system as in [1]. However, the exponential operation requires special attention.

From (2), we can see that the exponential values in the Gabor filter generation are determined by ϕ, δ_x and δ_y. Fig. 4 shows the impacts of parameter selections on the Gabor filtering output.

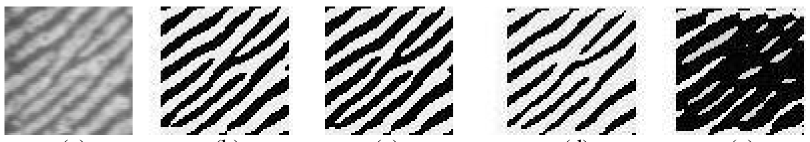


Fig. 4. Binarized Gabor filtering output under different parameter selections

Table 2. Gabor Filter parameters selection for Fig. 4

Image	δ_x	δ_y	ϕ	f
(b)	4.0	5.0	0.57	0.14
(c)	4.0	4.0	0.57	
(d)	3.5	5.0	0.57	
(e)	4.0	5.0	0.37	

Fig. 4a is the original fingerprint image block. Its local orientation and frequency are estimated to be 0.57 and 0.14, respectively. In Fig. 4b, δ_x and δ_y are carefully selected so that the filtered image best reflects the ridge structure of the original fingerprint image. Fig. 4c ~ Fig. 4e are the filtering outputs with deviations in one of the parameters as listed in Table 2. We can see that the effectiveness of Gabor filtering is very sensitive to ϕ and δ_x, but relatively less sensitive to δ_y. This is not difficult to understand. Parameter ϕ reflects the intrinsic property of ridge orientation for the fingerprint image block. Parameter δ_x influences both the noise smoothing intensity and the contrast between the ridge and valley in the filtering output while δ_y only determines the intensity of the noise smoothing along the local ridge orientation.

In our implementation, we carefully select δ_x according to the property of specific fingerprint database. Then, we let

$$\delta_y = \delta_x = \delta \tag{5}$$

In this way, equation (2) becomes

$$g(x, y; \phi, f) = \exp\left\{-\frac{1}{2\delta^2}\left[x^2 + y^2\right]\right\} \times \cos(2\pi f x_\phi) \tag{6}$$

We find that equation (6) has a very nice property that the exponential part is not affected by the choice of value for ϕ, implying that if once the value of δ and the size of the Gabor filter are determined for an image block, the exponential part will remain invariant no matter what the local orientation might be. In our implementation, a lookup table storing these exponential values is built inside our enhancement system.

3.3 Filtering

An enhanced image is obtained by performing 2D convolutions between the generated Gabor kernels and the corresponding image blocks. Suppose the size of an image I is $N \times N$ and the size of the Gabor filter matrix, G, is $2w+1$. According to the mathematical definition of a 2D convolution, the computational complexity of a direct 2D convolution implementation is $N^2(2w+1)^2$, a very time consuming operation. To speed up this step, we require a fast convolution algorithm. Existing fast convolution algorithms can be divided into mainly two categories: algebraic and domain transformation [17]. The domain transformation approaches always lead to extensive exponential operations that should be avoided in embedded systems. Therefore, we choose the algebraic approach.

An efficient fast 2D convolution algorithm takes advantage of the separability of G, a Gabor kernel [13]. According to [14], G can always be expressed in the form of $G = UQV^T$, where U and V are two orthogonal matrices and Q is a diagonal matrix with nonnegative diagonal entries. The number of positive diagonal entries of Q is equal to the rank of G. The process for decomposing a matrix into such a form is called Singular Value Decomposition (SVD). Suppose G has a rank of r, we can decompose G into the form

$$G = [U_1 \quad U_2 \quad \ldots] \times \begin{bmatrix} \mu_1 & & & 0 & \\ & \mu_2 & & & \\ & & \cdots & & \\ & & & \mu_r & \\ 0 & & & 0 & \\ & & & & \cdots \end{bmatrix} \times ([V_1 \quad V_2 \quad \ldots])^T \tag{7}$$

where μ_i are positive numbers and U_i and V_i are column vectors. Therefore, we have

$$G = \sum_{i=1}^{r} \mu_i \times \left(U_i \times V_i^T\right) \tag{8}$$

Applying the associability of the convolution, we have

$$I * G = \sum_{i=1}^{r} \mu_i \times \left((I * U_i) * V_i^T\right) \tag{9}$$

implying that a 2D convolution can always be decomposed into combinations of 1D convolutions. The computational complexity of (9) is $2rN^2(2w+1)$. Since N is fixed, the efficiency improvement can be expressed by

$$1 - \frac{2r}{2w+1} \tag{10}$$

If r is much smaller than $2w+1$, considerable computational effort can be saved. It's easy to prove that the even symmetric Gabor Filter matrices under the mathematical constraint (5) always have a rank no bigger than 2. In our implementation, we use $w = 5$. So, according to (10), ideally we can achieve a computation complexity decrease of 64%.

A SVD algorithm using fixed-point arithmetic based on the traditional GOLUB-REINSCH algorithm [15] is used in our implementation. As the range of values represented in our fixed-point arithmetic system is more restrictive than its floating-point counterparts, overflows and underflows in the SVD process can occur more frequently. Underflows can lead to some inaccuracy in the results if we replace the underflow values by zero. On the other hand, the consequence of overflow cannot be neglected. An accuracy check step is added at the end of the decomposition. If the error of the SVD result is estimated to be bigger than a certain tolerance value, the algorithm is regarded as a failure and a direct convolution is adopted instead. We used 53454 Gabor kernel matrices of different parameters to test our SVD algorithm and found only 1552 failure cases (2.9%).

4 Complexity Analysis

Mathematical derivation shows that SVD can considerably decrease the computational complexity of the convolution process. Ideally, we can achieve computational improvement of 64% in the convolution step by using SVD. However, such an achievement has not taken into consideration of the computational complexity of the SVD algorithm itself. In our fingerprint image enhancement, one SVD has to be done for each image block. Therefore for a complete fingerprint image, the computational complexity of SVD using Golub-Reinsch algorithm is roughly equal to ($8n^3 + 6n^2$) x B add+multiply operations, where n is the size of the Gabor filter kernel and B is the number of the image blocks. The complexity improvements under several different image sizes are shown in Table 3 in which we have assumed that the complete image is to be filtered and ignored the extra complexity which might arise due to the failure of the SVD process. The first two rows are the practical situations in which we really face in our experiments. The last row is only an imaginary situation to show the tendency. We can see that, given a fixed number of blocks, the bigger the Gabor kernel and the image, more significant performance improvements can be achieved through SVD.

Table 3. Computational complexity comparison of the convolution process with/without SVD

Image Size	Block Size	Block Number	Gabor Kernel Size	Complexity Analysis (Million add+multiply Operations)		
				No SVD	SVD	Improvement
256x256	16x16	256	11x11	7.9	6.9	13%
512x512	32x32	256	11x11	31.7	16.4	48%
1024x1024	64x64	256	17x17	303.0	83.3	72.5%

5 Experiments and Results

The purposes of our experiments are to prove that 1) considerable speed improvements 2) effectiveness of the fingerprint enhancement can be attained in our implementation.

Two fingerprint databases were used in our experiments. One is the database (FP) set up in our lab with 1149 fingerprints from 383 individuals; 3 for each: Image size is 256 x 256. The other is a subset of the NIST4 databases with 1000 fingerprints from 500 individuals; two for each: Image size is 512 x 512.

Two fingerprint image enhancement program modules were developed for comparison purpose. One adopts our fixed-point arithmetic based implementation (module A) and the other uses the traditional floating-point based implementation (module B). In both modules, the size of the Gabor kernel size is set to 11 and the block size is set to 16 for FP and 32 for the NIST4 subset. The values of δ are set to 2.0 and 4.0 for FP and the NIST4 subset respectively.

The first part of the experiments is to test the speed of our implementation. The experiments were conducted in a PDA equipped with a 206MHz StrongARM processor. 30 fingerprint images were taken from each fingerprint database. Fingerprint image enhancement was applied to these fingerprint images using module A and module B respectively. The timings of the image enhancement process for module B and module A are listed in Table 4 and Table 5 respectively. Compared to the traditional floating-point based implementation, our fixed-point based implementation has achieved considerable improvements in processing speed. In our embedded verification system, the average computational time for enhancing one fingerprint image is near to that of the minutiae extraction [1]. Also we notice that for the NIST4 fingerprint images, the use of SVD speeds up the convolution process considerably.

Table 4. The runtime of the fingerprint image enhancement in the embedded system using floating point based implementation

Database Used	**Normalization (seconds)**	**Orientation (seconds)**	**Frequency & ROI (seconds)**	**Local Tuning (seconds)**	**Filtering (seconds)**	**Total (seconds)**
FP	1.71	0.03	0.67	0.60	70.72	73.73
NIST4	6.86	0.11	1.47	2.55	275.81	286.80

Table 5. The runtime of the fingerprint image enhancement in the embedded system using fixed point based implementation

Database Used	**Normalization (seconds)**	**Orientation (seconds)**	**Frequency & ROI (seconds)**	**Local Tuning (seconds)**	**Filtering (seconds)**		**Total (seconds)**	
					No SVD	**SVD**	**No SVD**	**SVD**
FP	0.03	0.03	0.09	0.08	0.60	0.67	0.83	0.90
NIST4	0.12	0.11	0.31	0.34	2.48	1.87	3.36	2.75

The second part of the experiment is to test the verification performance after the fingerprint image enhancement. Because of the storage limitation of the embedded system (32MB), this experiment was simulated in a PC Linux system.

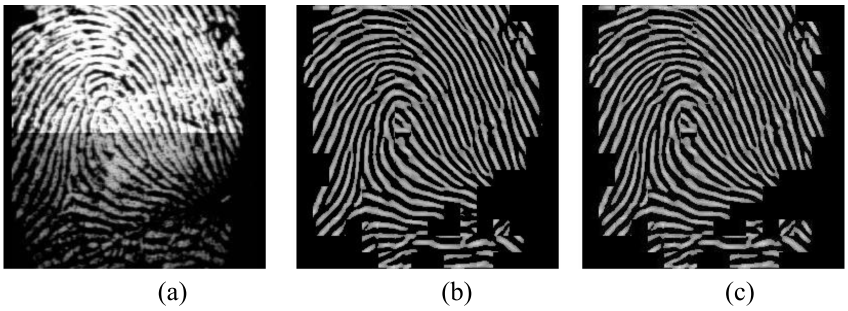


Fig. 5. Example of enhancement result: (a) Original Fingerprint Image; (b) Enhanced Image using module B; (c) Enhanced Image using module A

The experiment was performed in three steps. First the verification was performed directly on all fingerprints without any enhancement. Then, the images were enhanced using module B, and finally module A was employed for the enhancement. Samples of enhanced images using module A and module B are shown in Fig. 5.

The receiver operating curves (ROC) obtained from applying the experiments to FP and the NIST4 subset are shown in Fig. 6 and Fig. 7. Both curves show that the effectiveness of enhancement using our fixed-point arithmetic based implementation is basically similar to its floating-point arithmetic counterpart.

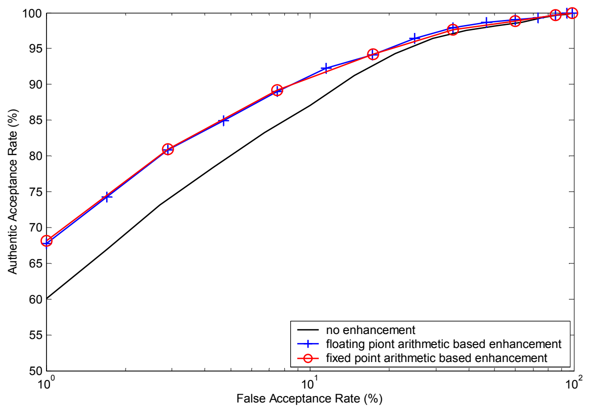


Fig. 6. Receiver Operating Curves (ROC) showing the comparison of the enhancement efficiency using module A and module B in the FP database

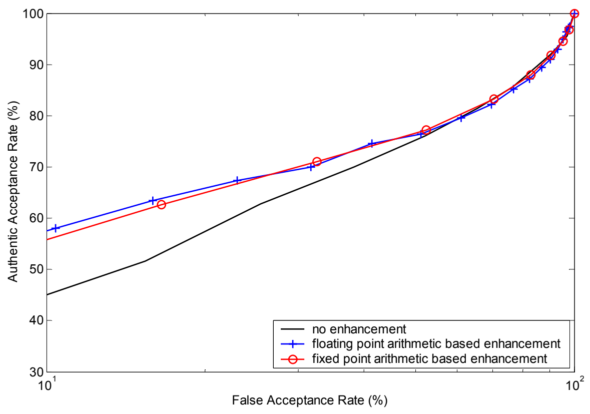


Fig. 7. Receiver Operating Curves (ROC) showing the comparison of the enhancement efficiency using module A and model B in the NIST4 subset database

6 Conclusion

We have proposed an efficient fingerprint enhancement implementation for mobile embedded systems. In our work, we set up a special constraint on the selection of the parameters for the Gabor kernel so that the complexity of the Gabor kernel generation can be significantly reduced. This parameter selection scheme also leads to the property of very low rank of the generated Gabor kernel matrices whose SVD can be easily accomplished. Experiment results show that our new implementation has achieved a considerable speed improvement, enabling fingerprint image enhancement to be accomplished in modern mobile devices.

We can observe from the experiment results that the influence on the processing speed using SVD can be negative for small fingerprint images. The main reason is that the Golub-Reinsch based SVD algorithm used in our implementation is a SVD algorithm which does not take advantage of low rank properties of the Gabor kernel matrices under the mathematical constraint stated in (5). To develop a specific SVD algorithm that makes use of such computationally efficient algebraic property will be our next task.

Acknowledgement

This work was partially supported by the Hong Kong Research Grants Council Project 2300011, “Towards Multi-Modal Human-Computer Dialog Interactions with Minimally Intrusive Biometric Security Functions”.

References

1. T. Y. Tang, Y. S. Moon, K. C. Chan: Efficient Implementation of Fingerprint Verification for Mobile Embedded Systems using Fixed-point Arithmetic, Proceedings of the 2004 ACM Symposium on Applied Computing, pp. 821-825, March, 2004
2. K. C. Chan, Y. S. Moon, P. S. Cheng: Fast Fingerprint Verification Using Sub-regions of Fingerprint Images, Circuits and Systems for Video Technology, IEEE Trans., Vol. 14, Issue 1, pp. 95-101, Jan. 2004
3. H. Lin, Y. Wan, A. K. Jain, Fingerprint image enhancement: algorithm and performance evaluation, Pattern Analysis and Machine Intelligence, IEEE Trans., Vol. 20, Issue 8, pp.777, Aug. 1998
4. J. Cheng, J. Tian, H. Chen, Q. Ren, X. Yang: Fingerprint Enhancement Using Oriented Diffusion Filter, AVBPA 2003, LNCS 2688, pp.164-171
5. X. Luo, J. Tian: Knowledge Based Fingerprint Image Enhancement, Proceedings of 15th International Conference on Pattern Recognition, Vol. 4, pp.783 – 786, Sept. 2000
6. W. P. Zhang; Q. R. Wang, Tang, Y.Y.: A wavelet-based method for fingerprint image enhancement, Proceedings of International Conference on Machine Learning and Cybernetics 2002, Vol. 4, pp.1973 – 1977, Nov. 2002
7. D. Maio, D. Maltoni: Direct gray-scale minutiae detection in fingerprints, Pattern Analysis and Machine Intelligence, IEEE Trans., Vol. 19, Issue 1, pp.27 –40, Jan. 1997
8. Y. S. Moon, F. T. Luk, T. Y. Tang, K.C. Chan, C. W. Leung: Fixed-Point Arithmetic for Mobile Devices—A Fingerprint Verification Case Study, Proceedings of the SPIE 2002 Seattle, Advanced Signal Processing Algorithms, Architectures, and Implementations XII, Vol. 4791, pp. 144 -149

9. M. Willems, V. Buersgens, H. Keding, H. Meyr: FRIDGE: An Interactive Fixed-Point Code Generation Environment for HW/SW CoDesign, Proceeding of the International Conference on Acoustics, Speech, and Signal Processing '97, Apr. 1997
10. J. Yang, L. Liu, T. Jiang, Y. Fan: A modified Gabor filter design method for fingerprint image enhancement, Pattern Recognition Letters, v.24, pp.1805-1817, 2003
11. A. K. Jain, F. Farrokhnia: Unsupervised texture segmentation using Gabor filters, Proceedings of IEEE International Conference on Systems, Man and Cybernetics, pp.14–19, Nov. 1990
12. S. Greenberg, M. Aladjem, D. Kogan, I. Dimitrov: Fingerprint image enhancement using filtering techniques, Proceedings of 15th International Conference on Pattern Recognition, Vol. 3, pp.322 – 325, Sept. 2000
13. R.M. Haralick, L.G. Shapiro: Computer and Robot Vision, Vol. I, Addison-Wesley, pp.298-299, 1992
14. K. E. Atkinson: An Introduction to Numerical Analysis, John Wiley & Sons, pp. 408, 1978
15. G. H. Golub, C. Reisch: Singular value decomposition and least squares solutions, Handbook for Automatic Computation, Vol. 2 (Linear Algebra), pp.134—151, New York: Springer-Verlag, 1971
16. J. G. Daugman: Uncertainty Relation for Resolution in Space, Spatial-Frequency and Orientation Optimized by Two-Dimensional Visual Cortical Filters, J. Optical Soc. Am., Vol. 2, pp.1160-1169, 1985
17. R. Tolimieri, M. An, Chao Lu: Algorithms for discrete Fourier transform and convolution, published by New York, Springer, 1997
18. Y. S. Moon, C. C. Leung, K. H. Pun: "Fixed-point GMM-based Speaker Verification over Mobile Embedded System", Proceedings of ACM SIGMM 2003 Multimedia Biometrics Methods and Applications Workshop, pp. 53-57, Nov. 2003