Accurate and Efficient Computation of Gabor Features in Real-Time Applications

Gholamreza Amayeh †, Alireza Tavakkoli ‡ and George Bebis†

† Computer Vision Laboratory, University of Nevada, Reno
‡ Computer Science Department, University of Houston–Victoria

Abstract. Gabor features are widely used in many computer vision applications such as image segmentation and pattern recognition. To extract Gabor features, a set of Gabor filters tuned to several different frequencies and orientations is utilized. The computational complexity of these features, due to their non-orthogonality, prevents their use in many real-time or near real-time tasks. Many research efforts have been made to address the computational complexity of Gabor filters. Most of these techniques utilize the separability of Gabor filters by decomposing them into 1-D Gaussian filter. The main issue in these techniques is the efficient pixel interpolation along the desired direction. Sophisticated interpolation mechanisms minimize the interpolation error with the increased computational complicity. This paper presents a novel framework in computation of Gabor features by utilizing a sophisticated interpolation scheme – quadratic spline – without increasing the overall computational complexity of the process. The main contribution of this work is the process of performing the interpolation and the convolution in a single operation. The proposed approach has been used successfully in real-time extraction of Gabor features from video sequence. The experimental results show that the proposed framework improves the accuracy of the Gabor features while reduces the computational complexity.

1 Introduction

Recently, computer scientists have become interested in modeling the human vision systems [1]. It is explained by neuroscientists [2] that receptive fields of the human vision system can be represented as basis functions similar to Gabor filters. In NeoCortical Simulators(NCS), the response profile of neurons in visual cortex area of the human brain is modeled by Gabor features. In the latest version of the NCS (version 5.0), the robot's eye (a tracking pan-tilt-zoom camera) captures the video images from the real world [3]. Then, Gabor features of these images are extracted and uploaded to the brain simulator running on a cluster of computers, executing a pre-specified spiking brain architecture. Real-time extraction of these features from video images (30 frames per second) is important in order to avoid small delays which slow down the entire system. Moreover, inaccurate features can trigger inappropriate neuron regions. As a result, accurate and efficient extraction of Gabor features from video sequences is a crucial task.

Gabor features are based on Gabor filter responses to a given input image. The responses over the image are calculated for a set of filters – *a bank* – tuned to various orientations and frequencies. The most straightforward technique to conduct the filtering operation is by performing the convolution in the spatial domain. The complexity of convolution depends directly on the size of the convolution mask. The mask in this case is the Gabor filter. The complexity of calculating the filter response for one point is $O(M^2)$, where M is the width and height of the mask. If the filtering is done on the entire image of size $N \times N$, the complexity becomes $O(M^2N^2)$.

One trivial solution to reduce the computational complexity is to perform the filtering process in the frequency domain [4]. In this approach, the image is first converted to the frequency domain using the Fast Fourier Transform (FFT). Afterwards the FFT transformed image is multiplied by a FFT transformed Gabor filter. Finally, the responses are converted back to the spatial domain using the inverse FFT. For an image of size $N \times N$, the computational complexity of this approach becomes $O(N^2 \log N)$ with a constant multiplier [4]. One of the issues with this method is the fact that the generic FFT formulation is limited to signals of length 2^n. Moreover, the memory requirement of this approach is very high.

Many research efforts have been made to significantly improve the computational complexity of Gabor filtering [5–8]. Nestares *et al.* in [6] improved the standard convolution with Gabor filters by utilizing the separability of Gabor filters. Ranganathan *et al.* in [5] used symmetry and anti-symmetry characteristics of Gabor filters to reduce their computational complexity. These convolution improvements can reduce the computational complexity of the Gabor filter from $O(M^2N^2)$ to $O(2MN^2)$. Compared to FFT filtering complexity of $O(N^2 \log N)$ it is evident that these techniques are beneficial when $M < \log N$. The main issue is that these methods can be applied only to certain configurations (e.g. $\theta = k\frac{\pi}{4}$, $k \in Z$), making them merely special cases.

Recently Areekul *et al.* in [8] generalized separable Gabor filters to any orientation. Their method uses three steps. The first step is to define and interpolate consecutive sequences of pixels to form a new image along selected convolutional orientations and their perpendicular directions. They employed an interpolation technique with the least expensive complexity – the linear interpolation of the two nearest pixels. Secondly, two continuous 1-D Gabor filters with suitable parameters are generated and re-sampled with uniform space between pixels. This task resembles the image re-sampling from the first step. Finally, separable convolutions can be performed along any selected orientation using these tessellated and interleaved patterns. In the best cases when $\theta = k\frac{\pi}{4}$, $k \in Z$ the computational complexity is $O(2MN^2)$ [8]. However, for an arbitrary orientation the re-sampling process plays a critical role if the required pixel is not on the sampling grid. As a result, in the worst case scenarios the computational complexity reaches $O(6MN^2)$. In this method, the main issue is interpolation error resulting in less accurate Gabor features. The accuracy can be improved by employing more sophisticated interpolation schemes (e.g. quadratic spline). Unfortunately,

such sophisticated techniques drastically increase the computation cost of the first step – up to 4 times.

In this paper, a new framework is presented to generalize separable Gabor filters for any orientation by integrating the interpolation and the convolution processes in a single step. The proposed approach employs a sophisticated interpolation method – quadratic spline. This, results in a low interpolation error without any increase in the computational complexity. As it was mentioned, to extract Gabor features, a filter bank containing $Q \times S$ Gabor filters in Q directions and S scales is utilized. In this study, the computation complexity of Gabor features is further reduced by applying 1D filters in specific direction for all scales. This paper is organized as follows: In section 2, a review of the Gabor filter is presented. In section 3, the details of separable Gabor filters are described. Section 4 discusses the integration of the interpolation and the convolution processes in a single step. Section 5 shows experimental results of the proposed approach and compares our method with the state-of-the-art. Finally, Section 6 concludes this work and discusses future directions of this study.

2 Gabor Filters

The Gabor filter is a product of an elliptical Gaussian in any rotation and a complex exponential function representing a sinusoidal plane wave [9]. The sharpness of the filter is controlled on the major and minor axes by σ_x and σ_y, respectively. The filter response in spatial domain can be expressed by the following equation [9]:

$$g(x, y, f, \theta) = e^{-\frac{1}{2}\left(\frac{x_\theta^2}{\sigma_x^2}+\frac{y_\theta^2}{\sigma_y^2}\right)} \times e^{j2\pi f x_\theta} \tag{1}$$

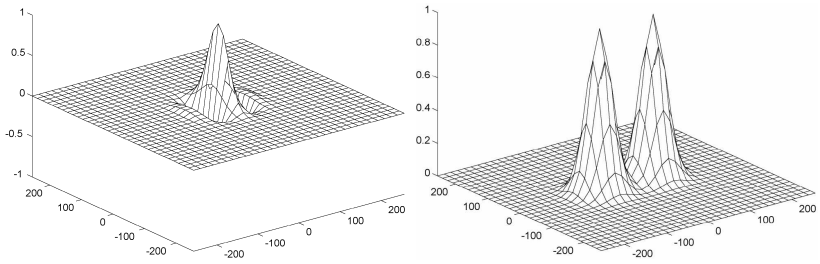
where f is the frequency of the sinusoidal plane wave, θ is the orientation of the Gabor filter, σ_x is the sharpness along the major axis, σ_y is the sharpness along the minor axis, $x_\theta = x\cos\theta + y\sin\theta$ and $y_\theta = -x\sin\theta + y\cos\theta$. In most applications, the real part of the filter's impulse response (namely even-symmetric Gabor filter) is considered. As a result, the equation 1 can be rewritten as [9]:

$$g(x, y, f, \theta) = e^{-\frac{1}{2}\left(\frac{x_\theta^2}{\sigma_x^2}+\frac{y_\theta^2}{\sigma_y^2}\right)} \times cos(2\pi f x_\theta) \tag{2}$$

The normalized Gabor filter in the frequency domain can be represented by [9]:

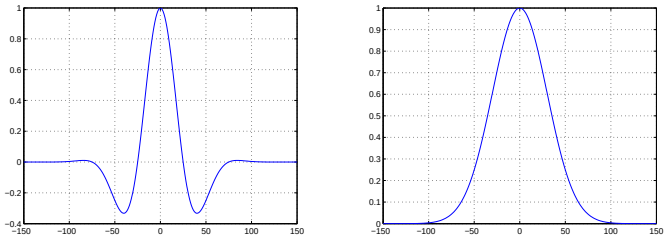
$$G(u, v, f, \theta) = \frac{1}{2\pi\sigma_u\sigma_v}\left[e^{-\frac{1}{2}\left(\frac{(u_\theta-u_0)^2}{\sigma_u^2}+\frac{(v_\theta-v_0)^2}{\sigma_v^2}\right)} + e^{-\frac{1}{2}\left(\frac{(u_\theta+u_0)^2}{\sigma_u^2}+\frac{(v_\theta+v_0)^2}{\sigma_v^2}\right)}\right] \tag{3}$$

where u_0, v_0 and $\sigma_{u,v}$ are equal to $\frac{2\pi\cos\theta}{f}$, $\frac{2\pi\sin\theta}{f}$ and $\frac{1}{2\pi\sigma_{x,y}}$, respectively. Also $u_\theta = u\cos\theta + v\sin\theta$ and $v_\theta = -u\sin\theta + v\cos\theta$. Figure 1 shows an even-symmetric Gabor filter in the spatial and frequency domains.



(a) (b)

Fig. 1. An even-symmetric Gabor filter with $\theta = \frac{\pi}{2}$, $f = 0.01$ and $\sigma_x = \sigma_y = 3$ in (a) spatial domain and (b) frequency domain.



(a) (b)

Fig. 2. Decomposing the Gabor filter in Fig. 1 into (a) a band-pass Gaussian filter, and (b) a low-pass Gaussian filter.

3 Separability of Gabor Filters

A filter g is called separable if it can be expressed as the multiplication of two vectors – $g_{row} \times g_{col}$. For separable filters the convolution can be performed separately with one dimensional filters g_{row} and g_{col}. Employing one dimensional filters decreases the two dimensional filter's computational complexity from $O(M^2N^2)$ to $O(2MN^2)$, where M and N are the width (and height) of the filter mask and the image, respectively.

According to the definition of separable filters, the Gabor filters are parallel to the image axes – $\theta = k\frac{\pi}{2}$, $k = 0, 1, ...$ – are separable. In separable Gabor filters, one of the 1-D filters is a sinusoidal function with a Gaussian envelope and the other one is a Gaussian envelope. For example, if $\theta = \frac{\pi}{2}$, equation (1) gives $x_\theta = x$ and $y_\theta = y$. Therefore, equation (2) can be rewritten as:

$$g(x, y, f, \theta) = g_{bp}(x, f) \times g_{lp}(y) = e^{-\frac{x^2}{2\sigma_x^2}} \cos(2\pi f x) \times e^{-\frac{y^2}{2\sigma_y^2}} \tag{4}$$

where g_{bp} is a 1D band-pass Gaussian filter, and g_{lp} is a 1D low-pass Gaussian filter as shown in figure 2.

Separable Gabor filters can be extended to work with $\theta = k\frac{\pi}{4}$ by going through the image along diagonal directions instead of the image axes [6]. In order to implement separable 2D Gabor filters in any direction it should be separated into 1-D low-pass and band-pass filters along the desired orientation and its perpendicular direction, respectively. However, the line formed by pixel sequences along an arbitrary direction $\theta \neq k\frac{\pi}{4}$ is not well defined due to square sampling grid pattern in an image. For example, if we draw a straight line in any chosen direction, it is difficult to pick consecutive pixels in order to form a straight line in that particular direction. Therefore, making Gabor filters separable in arbitrary directions needs a re-sampling process. The re-sampling should be performed to get an exact sequence of missing pixels on the desired orientations.

4 Integrating Interpolation and Convolution

In general, there are several ways to find approximate values for the re-sampled pixels. Linear interpolation, spline interpolation, or sinc interpolation are among the most widely used techniques in image re-sampling. In all of these schemes, there is a trade off between computational complexity and interpolation error.

Areekul *et al.* in [8] proposed a generalized separable Gabor filter for any orientation. Their method has two main steps. The first step is to interpolate consecutive sequences of pixels along an arbitrary direction. The second step performs a separable convolution along the direction. In order to reduce the computational complexity of the interpolation process in [8], missing pixels are linearly interpolated between their two nearest pixels. Although this interpolation scheme has a low complexity $O(2MN^2)$, it suffers from increased interpolation error. Employing more sophisticated interpolation schemes in this approach will increase the computational complexity significantly – e.g. $O(9MN^2)$ for quadratic spline interpolation.

We proposed a novel approach in performing the interpolation and convolution processes required to achieve a separable Gabor filter along an arbitrary direction. The main idea behind our framework is the integration of the interpolation and the convolution processes. To this end, we propose a technique to re-sample an image $f(x, y)$ by an interpolation kernel $k(x, y)$ and then convolve it by a convolution kernel $p(x, y)$. By performing this integration scheme the overall process saves one step by convolving the image by a kernel $q(x, y)$.

Let's define $f_i(x, y)$ to be the image after interpolation but before convolution:

$$f_i(x, y) = \sum_{x_1, y_1} f(x_1, y_1)k(x - x_1, y - y_1) \tag{5}$$

By convolving it with $p(x, y)$ we get the final result $f_p(x, y)$:

$$f_p(x, y) = \sum_{x_2, y_2} f_i(x - x_2, y - y_2)p(x_2, y_2) \tag{6}$$

Substituting equation (5) in (6) results in:

$$f_p(x,y) = \sum_{x_2,y_2} [\sum_{x_1,y_1} f(x_1,y_1)k(x-x_2-x_1, y-y_2-y_1)]p(x_2,y_2) \tag{7}$$

After regrouping the sums, we get:

$$\begin{aligned} f_p(x,y) &= \sum_{x_1,y_1} f(x_1,y_1)[\sum_{x_2,y_2} k(x-x_2-x_1, y-y_2-y_1)p(x_2,y_2)] \\ &= \sum_{x_1,y_1} f(x_1,y_1)q(x-x_1, y-y_1) \end{aligned} \tag{8}$$

Therefore, the final result – $f_p(x,y)$ – is the convolution of the image $f(x,y)$ by $q(x,y)$, where $q(x,y) = k(x,y) * p(x,y)$.

The 1D low-pass Gaussian filter – g_{lp} – in the direction θ can be generated with uniform displacement d. In this case d is related to θ by the following equation:

$$d = \begin{cases} \frac{1}{\cos\theta} & \text{if } |\cos\theta| \geq \frac{\sqrt{2}}{2} \\ \\ \frac{1}{\sin\theta} & \text{if } |\sin\theta| > \frac{\sqrt{2}}{2} \end{cases} \tag{9}$$

The same relation is true for the 1-D band-pass filter along the perpendicular direction as well. Therefore, the 1D low-pass and band-pass Gaussian filters become:

$$\begin{aligned} g_{bp}[n] &= e^{-\frac{(nd)^2}{2\sigma_x^2}} \cos(2\pi f n d) \\ g_{lp}[n] &= e^{-\frac{(nd)^2}{2\sigma_y^2}} \end{aligned} \tag{10}$$

When the orientation is $\theta = k\frac{\pi}{4}$ all of $g_{lp}[n]$ and/or $g_{bp}[n]$ are located on the sampling Cartesian grid. However, for an arbitrary direction (figure 3(a)) some of the low- and/or band- pass filtered pixels (i.e. the red circles) may be on the sampling Cartesian grid while others may not. For the pixels which do not lie on the sampling gird their pixel values need to be regenerated by a re-sampling process. We utilize a quadratic B-spline interpolation scheme to estimate these missing pixel values. Moreover, the proposed framework can be extended to other interpolation schemes as well. The B-spline re-sampling process is one of the most commonly used family of spline functions [10]. It can be derived by several self-convolutions of a basis function. Quadratic B-spline interpolation kernel, $k(r)$, can be presented by the following formula [11]:

$$k(r) = \begin{cases} \frac{1}{2}r^2 & \text{if } 0 < |r| \leq \frac{1}{2} \\ -r^2 + r + \frac{1}{2} & \text{if } \frac{1}{2} < |r| \leq 1 \\ \frac{1}{2}(1-r^2) & \text{if } 1 < |r| \leq \frac{3}{2} \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

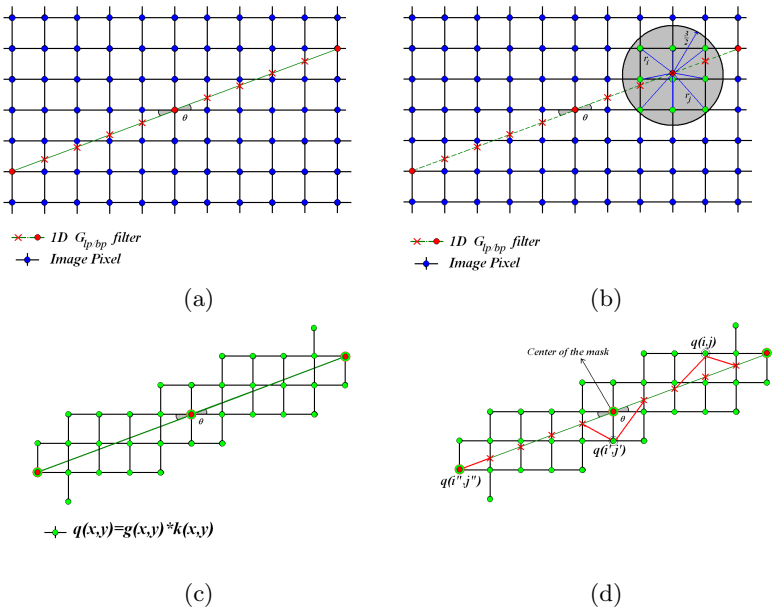


Fig. 3. (a) Some pixels in 1D filtering may not on sampling Cartesian grid; (b) Regenerating missing pixels by Quadratic B-spline interpolation; (c) Building the mask q which integrate 1D convolution and Quadratic B-spline interpolation; (d) Calculating the $q(i, j)$ coefficients.

where r is the distance of an estimated value from a pixel on the sampling Cartesian grid – figure 3(b).

Instead of performing interpolation and convolution separately, we can accomplish them in a single step by defining a mask q, shown in figure 3(c). This mask is the convolution of the interpolation kernel $k(r)$ and the 1D low- and/or band-pass Gaussian filters – g_{lp} and/or g_{bp}. The size of q is almost $3 \times M$ in the worst case scenario – figure 3(c). As a result, the asymptotic computational complexity of our approach is $O(6M^2N^2)$.

For the implementation purpose, we find all Cartesian grids whose minimum distance to $g_{mis}[n]$ (the red crosses in figure 3(d)) is less than $\frac{3}{2}$. Then we calculate their coefficients by the following formula:

$$q(i,j) = \sum_{l=-\frac{M}{2}, |r_l| \leq \frac{3}{2}}^{\frac{M}{2}} k(r_l) g_{mis}[n_l] \tag{12}$$

Some $g_{lp}[n'']$ and/or $g_{bp}[n'']$ positions may be located on the sampling Cartesian grid of the mask q – the $q(i'', j'')$ in figure 3(d). In these cases we need to update the $q(i'', j'')$ by adding $g_{lp}[n'']/g_{bp}[n'']$ to it.

5 Experimental results

In this section some experiments have been conducted in order to evaluate the performance and efficiency of the proposed framework. The computational complexity of our approach is compared to the traditional techniques in the literature. To extract Gabor features from an acquired image, a Gabor filter bank containing $P \times Q$ Gabor filters in different directions $\theta \in \{\theta_1, \theta_2, ..., \theta_Q\}$ and different frequencies $f \in \{f_1, f_2, ..., f_P\}$, is utilized. From equation (**??**), in separable Gabor filters, the frequency only affects the 1-D band-pass Gaussian filter. Therefore, in a practical implementation, the 1-D low-pass Gaussian filter is computed once for all P Gabor filters in a specific direction θ_i with different frequencies.

Table 1. Comparison of different methods in terms of memory requirement and computational complexity.

Techniques	Memory space	Computation cost for a Gabor filter	Computation cost for a Gabor bank
Traditional method	$O(N^2)$	$O(M^2N^2)$	$O(PQM^2N^2)$
Filtering in frequency domain	$O(3N^2)$	$O(12N^2 \log N)$	$O(12PQN^2 \log N)$
Separable method[8]	$O(2N^2)$	$O(6MN^2)$	–
Proposed method	$O(2N^2)$	$O(6MN^2)$	$O(3(P+1)QMN^2)$

Table 1 shows the computational complexity and the memory requirement of our proposed approach compared to the traditional method of filtering in frequency domain, and the separable technique proposed in[8]. In this experiment we use a specific direction and frequency and a Gabor filter bank containing $P \times Q$ filters. Although the traditional method in spatial domain is the least expensive technique in terms of memory requirement, it is the most computationally complex. As compared to filtering in the frequency domain, $O(12N^2 \log N)$, the spatial domain convolution with separable filters is beneficial when $M < 2 \log N$. Moreover, the generic FFT formulation is limited to working with signals of length 2^n. From table 1, however, the proposed method employs a sophisticated interpolation scheme based on Quadratic B-spline. Compared to the method in [8], it is evident that our approach does not increase the computational complexity and the memory requirements.

In this experiment, we have employed different methods to extract Gabor features from a video sequence with dimensionality of 640×480 and the rate of 30 frames per second. Gabor features were extracted using 12 Gabor filters in 4 different directions $\theta \in \{\frac{\pi}{8}, \frac{3\pi}{8}, \frac{5\pi}{8}, \frac{7\pi}{8}\}$, and 3 different scales $f \in \{1, 3, 5\}$. All methods have been implemented in C/C++ on a 64-bit machine with 2G byte RAM. In the case of filtering in frequency domain, since the dimensionality of

Table 2. Average number of filtered video frames by a Gabor filter bank (12 filters) in one second using different Gabor filtering methods.

Techniques	$\overline{Speed}$ (fps)
Traditional method	1.1
Filtering in frequency domain	12.8
Separable method[8]	30
Proposed method	30

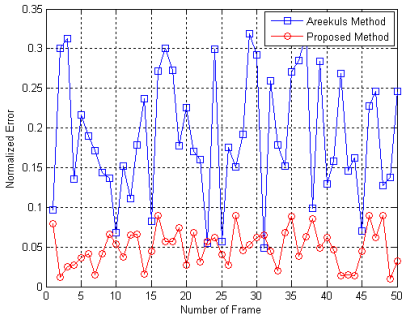


Fig. 4. Comparison of Areekul's method [8] and proposed method in terms of accuracy for the first 50 frames od video sequence.

video frames were not exactly the length of 2^n, we used FFTW3 library [12]. This technique provides fast solutions for the signal lengths that the original FFT is not suitable for. Table 2 shows that the separable Gabor filter has the best performance among others and, as expected, the traditional method has the worst one. To evaluate the accuracy of both separable approaches, we calculated the normalized error E:

$$E = \frac{\sum_x \sum_y |\dot{f}(x,y) - \ddot{f}(x,y)|^2}{\sum_x \sum_y \dot{f}(x,y)^2} \tag{13}$$

where $\dot{f}$ is the filtered image by traditional method in spatial domain, and $\ddot{f}$ is the filtered image by one of the separable Gabor techniques. Figure 4 shows the average normalized error for all direction and frequencies for the first 50 frames. As you can see, our proposed method has significantly reduced the interpolation error by employing Quadratic B-spline interpolation technique. As it was expected, the method in [8] has considerable error due to the use of linear interpolation of the two nearest pixels.

6 Conclusion

Fast and efficient computation of Gabor features from video frames have become the focus of recent studies of the functionalities of the human brain and visual cognitive systems. The issue of efficient calculation of Gabor filters is of particular interest since the directional properties of these filters makes them inseparable along arbitrary direction.

In this paper we proposed a novel technique to integrate the interpolation and the convolution processes of the Gabor filter. This integration of the two processes makes the 2-D Gabor filter separable along any direction. Moreover, the integration performs the two processes as a single step in real-time. Therefore, our interpolation process employs a sophisticated interpolation technique in order to increase its accuracy while performing the entire process in real-time.

References

1. Serre, T., Wolf, L., Poggio, T.: Object recognition with features inspired by visual cortex. IEEE Conf. on Computer Vision and Pattern Recognition **2** (2005) 994–1000
2. Olshausen, A.B., Field, D.J.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Intl. Jurnal of Nature **381** (1996) 607–609
3. Goodman, P., Zou, Q., Dascalu, S.: Framework and implications of virtual neurorobotics. Frontiers in Neuroscience **2** (2008) 123–128
4. Bracewell, R.N., ed.: The Fourier Transform and Its Applications. 3rd edition. McGraw-Hill (2000)
5. Ranganathan, N., Mehrotra, R., Namuduri, K.: An architecture to implement multiresolution. Intl. Conf. on Acoustics, Speech, and Signal Processing **2** (1991) 1157–1160
6. Nestares, O., Navarro, R., Portilla, J., Tabernero, A.: Efficient spatial-domain implementation of a multiscale image representation based on gabor functions. Journal of Electronic Imaging **7** (1998) 166–173
7. Areekul, V., Watchareeruetai, U., Tantaratana, S.: Fast separable gabor filter for fingerprint enhancement. Intl. Conf. on Biometric Authentication (2004) 403–409
8. Areekul, V., Watchareeruetai, U., Suppasriwasuseth, K., Tantaratana, S.: Separable gabor filter realization for fast fingerprint enhancement. IEEE Intl. Conf. on Image Processing **3** (2005) 253–256
9. Daugman, J.: Uncertainty relation for resolution in space, spatial-frequency, and orientation optimized by two-dimensional visual cortical filters. Journal of Optical Socity America **2** (1985) 160–169
10. Hou, H.S., Andrews, H.C.: Cubic splines for image interpolation and digital filtering. IEEE Tran. on Acoustic, Speech and Signal Processing **26** (1978) 508–517
11. Dodgson, N.A.: Quadratic interpolation in image resampling. IEEE Tran. on Image Processing **6** (1997) 1322–1326
12. Frigo, M., Johnson, S.: The design and implementation of fftw3. Proceedings of IEEE **93** (2005) 216–231