Computational Foundations of Image Interpolation Algorithms

Tinku Acharya[1], Ping-Sing Tsai[2]

[1]Avisere Inc., Tucson, Arizona
&
Department of Electrical Engineering
Arizona State University, Tempe, Arizona

[2]Department of Computer Science, University of Texas
– Pan American

Abstract

Image interpolation is an important image processing operation applied in diverse areas ranging from computer graphics, rendering, editing, medical image reconstruction, to online image viewing. Image interpolation techniques are referred in literature by many terminologies, such as image resizing, image re-sampling, digital zooming, image magnification or enhancement, etc. Basically, an image interpolation algorithm is used to convert an image from one resolution (dimension) to another resolution without loosing the visual content in the picture. Image interpolation algorithms can be grouped in two categories, non-adaptive and adaptive. The computational logic of an adaptive image interpolation technique is mostly dependent upon the intrinsic image features and contents of the input image whereas computational logic of a non-adaptive image interpolation technique is fixed irrespective of the input image features. In this paper, we review the progress of both non-adaptive and adaptive image interpolation techniques. We also proposed a new algorithm for image interpolation in discrete wavelet transform domain and shown its efficacy. We describe the underlying computational foundations of all these algorithms and their implementation techniques. We present some experimental results to show the impact of these algorithms in terms of image quality metrics and computational requirements for implementation.

Keywords: DWT, interpolation, filtering, sampling, scaling.

1. Introduction

We are in the midst of a visually enchanting world. Image processing technologies play central role to process visual information in order to make it suitable for multimedia applications and visual perception [1]. In this era of Internet and multimedia communication, sizing and resizing of images, to make it particularly suitable for viewing, transmission, downloading, sharing, editing, and further processing, became very vital. Image interpolation became a versatile and widely used tool in image processing today because of its numerous applications in diverse areas ranging from computer graphics, rendering, editing, medical image reconstruction, to online image viewing to name a few. Image interpolation techniques are referred in literature by many terminologies, such as image resizing, image re-sampling, digital zooming, image magnification, image enhancement, etc. Basically, image interpolation algorithms convert or resize a digital image from one resolution (dimension) to another resolution without loosing the visual content in the picture. Image interpolation is part of many commercial image processing tools or freeware graphic viewers such as Adobe® Photoshop® CS2 software [2], IrfanView [3], etc. Authors felt that a concise discussion on the techniques for image interpolation in a review nature will be helpful for the researchers, developers, and practitioners.

A static image is a two-dimensional spatially varying signal. This needs to be digitized in order to process it by a digital computer. In digital imagery, this two-dimensional spatially varied signal is *sampled* based on Nyquist criteria and then intensity of each sampled point is *quantized* to discrete valued integer numbers. Each such sampled and digitized value is called a pixel value of the digital image [1]. The resolution or dimension of the digital image is the number of sampled points of the two-dimensional signal. In case of color imagery, each pixel contains three intensity values representing the contribution of three primary colors Red, Green, and Blue to form the color of a particular pixel point in the image. As a result, a color image has primarily three such two-dimensional planes representing red, green, and blue components of the image pixels.

When the image is interpolated from a higher resolution to a lower resolution, it is traditionally called image *down-scaling* or *down-sampling*. On the other hand, when the image is interpolated from a lower resolution to a higher resolution, it is referred as image *up-scaling* or *up-sampling*. Most of the image interpolation techniques in the literature have been developed by interpolating the pixels based on characteristics of local features such as edge information, nearest-neighbor criteria, etc. Image interpolation techniques can be broadly categorized into two categories – *non-adaptive*, and *adaptive* techniques. The principles of adaptive interpolation algorithms basically rely on the intrinsic image features or contents of the image and accordingly the computational logic is mostly dependent upon the intrinsic image features and contents of the input image. The non-adaptive algorithms do not rely upon the image features or its contents and the same computational logic is repeated in every pixel or group of local pixels irrespective of the image contents.

Image interpolation technique in the transformed domain is rare in the literature [4]. Discrete Wavelet Transform (DWT) is a versatile tool to modern digital image processing applications such as pattern recognition, image enhancement, image compression, image interpolation [1, 5, 6], etc. Based on some of the salient features of DWT, we have proposed a novel image up-sampling technique using the DWT. Apparently, computation of the proposed technique is non-adaptive in nature. It is interesting to note that the technique has the flavor of adaptive nature as well due to the edge characteristics of the wavelet subbands and its contents.

Organization of the paper is as follows. In section 2, we review some of the key non-adaptive image interpolation algorithms widely used in the literature such as *nearest-neighbor* replacement, *bilinear* interpolation, *bicubic* interpolation, and some widely used *digital filtering* based approaches. In section 3, we present some adaptive interpolation algorithms exploiting the intrinsic image features such as hue, edge information, etc. We introduced a new interpolation algorithm based on *discrete wavelet transform* in section 4. We have discussed the computational requirements of all these algorithms in section 5. In section 6, we present some experimental results in terms of the image quality metrics using all these *adaptive* and *non-adaptive* algorithms. We conclude the paper in section 7.

2. Non-Adaptive Algorithms for Image Interpolation

In non-adaptive image interpolation algorithms, certain computations are performed indiscriminately to the whole image for interpolation regardless of its contents. In this section, we will discuss the popularly used non-adaptive image interpolation techniques which are typically used in commercial image processing tools or freeware graphic viewer such as Adobe® Photoshop® CS2 software [2] and IrfanView [3].

2.1. Nearest Neighbor Replacement

The simplest interpolation method is just to replace the interpolated point with the nearest neighboring pixel. The only advantage of this approach is the simplicity and low computation. However, the resultant pixelization or blocky effect makes the image quality unacceptable for most high quality imaging applications.

2.2. Bilinear Interpolation

The bilinear interpolation can be considered as a weighted average of four neighboring pixel values. As shown in Fig. 1, the intensity value *I(x, y)* at the interpolated point *P* at *(x, y)* in the image can be estimated as

$$\begin{aligned} I(x,y) &= \frac{(x_2-x)(y_2-y)}{(x_2-x_1)(y_2-y_1)}I(x_1,y_1)+\frac{(x-x_1)(y_2-y)}{(x_2-x_1)(y_2-y_1)}I(x_2,y_1)+\frac{(x_2-x)(y-y_1)}{(x_2-x_1)(y_2-y_1)}I(x_1,y_2)+\frac{(x-x_1)(y-y_1)}{(x_2-x_1)(y_2-y_1)}I(x_2,y_2) \\ &= w_1 I(x_1,y_1) + w_2 I(x_2,y_1) + w_3 I(x_1,y_2) + w_4 I(x_2,y_2) \end{aligned}$$

where *I(x₁, y₁)*, *I(x₁, y₂)*, *I(x₂, y₁)*, and *I(x₂, y₂)* are the intensity values of the 4 neighboring pixels. From the above equation and Fig. 1, it is interesting to note that each weight is equivalent to the normalized area between the point of interest *P* and the diagonally opposite pixel. For example, (x_2, y_2) is diagonally opposed to (x_1, y_1) and weight for I *I(x₁, y₁)* is w_1.

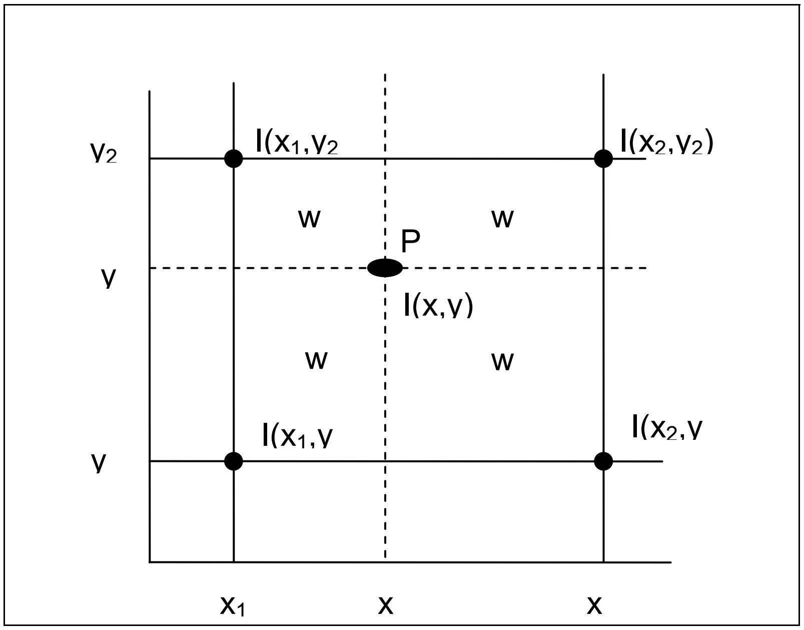


Fig. 1: Bilinear interpolation using 4 neighboring points.

2.3. Bicubic Interpolation

The bicubic interpolation uses sixteen (4×4) neighboring pixels for estimation. It approximates the local intensity values using a bicubic polynomial surface. The general form for a bicubic interpolation is as follows:

$$I_{xy} = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}x^i y^j = a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2 + a_{21}x^2y + a_{12}xy^2 + a_{22}x^2y^2 + a_{30}x^3 + a_{03}y^3 + a_{31}x^3y + a_{13}xy^3 + a_{32}x^3y^2 + a_{23}x^2y^3 + a_{33}x^3y^3 .$$

In order to do a bicubic interpolation within a grid square, one need to calculate the gradients (the first derivatives) in both the *x* and *y* directions and the cross derivative at *each* of the four corners of the square. This gives 16 equations that determine the 16 coefficients (a_{ij}), as explained in [7].

2.4. Filtering-based Techniques

The filtering-based methods are also known as re-sampling methods. As discussed in [8], re-sampling is the process of transforming discrete image pixels defined at one coordinate system to a new coordinate system of a different resolution. Frequently, the re-sampling technique is used to up-sample an image to enhance its resolution and appearance. As shown in Fig. 2, the re-sampling from one discrete signal $x[n]$ to a re-sampled signal $y[n']$ of a different resolution is computed as

$$y[n'] = \sum_{t=-w}^{w} h[t]x[n'-t] \, ,$$

where $h[t]$ is the interpolating function and *w* is the desired filtering window. The above equation can be considered as a simple filtering or convolution operation of *x*[*n*] with $h[t]$ as the discrete filter. Since digital image is a 2-D discrete function, 2-D convolution or filtering will be required. Direct 2-D convolution is computationally expensive. However, if we choose a 2-D separable function as $h[x,y] = h[x] \cdot h[y]$, we can accomplish the 2-D filtering by applying the 1-D filtering $h[x]$ row-wise in the image followed by column-wise 1-D filtering $h[y]$ of the intermediate result. As a result, the computation can be $O(N)$ compared to the *O*(*N*^{2}) for general 2-D filtering.

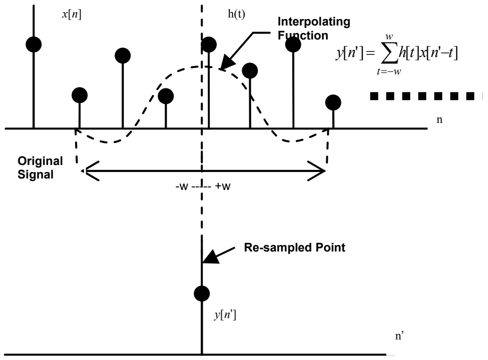


Fig. 2: Re-sampling/filtering of a single point.

Table I: Selected 2-D separable filters

$T(x) = \begin{cases} 1-\lvert x\rvert, & \lvert x\rvert < 1 \\ 0, & otherwise \end{cases}$ Linear-based (Triangle) filter [10]
$B(x) = \begin{cases} 0.75 - \lvert x\rvert^2 & ,\lvert x\rvert < 0.5 \\ 0.5 \times (\lvert x\rvert - 1.5)^2 & ,0.5 \le \lvert x\rvert < 1.5 \\ 0 & ,otherwise. \end{cases}$ Bell quadratic polynomial filter [11]
$H(x) = \begin{cases} 2\lvert x\rvert^3 - 3\lvert x\rvert^2 + 1 & ,\lvert x\rvert \le 1 \\ 0 & ,otherwise. \end{cases}$ Hermite cubic polynomial filter [12]
$M(x) = \frac{1}{6}\begin{cases} (12-9B-6C)\lvert x\rvert^3 + (-18+12B+6C)\lvert x\rvert^2 + (6-2B) & ,\lvert x\rvert < 1 \\ (-B-6C)\lvert x\rvert^3 + (6B+30C)\lvert x\rvert^2 + (-12B-48C)\lvert x\rvert + (8B+24C) & ,1 \le \lvert x\rvert < 2 \\ 0 & ,otherwise \end{cases}$ Michell cubic polynomial filters [10, 12]
$\beta^3(x) = \begin{cases} \frac{1}{2}\lvert x\rvert^3 - \lvert x\rvert^2 + \frac{2}{3} & ,\lvert x\rvert < 1 \\ -\frac{1}{6}\lvert x\rvert^3 + \lvert x\rvert^2 - 2\lvert x\rvert + \frac{4}{3} & ,1 \le \lvert x\rvert < 2 \\ 0 & ,otherwise. \end{cases}$ Cubic B-spline filter [10, 13]
$L3(x) = \begin{cases} \frac{\sin(\pi x)}{\pi x} \frac{\sin(\pi \frac{x}{3})}{\pi \frac{x}{3}} & ,\lvert x\rvert < 3 \\ 0 & ,otherwise. \end{cases}$ Lanczos3 sinc-based filter [14]

In this study, we selected six different 2-D separable filters widely used in computer graphics applications [9] for implementation and comparison of experimental results. The kernel functions of these filters are shown in Table I.

3. Adaptive Algorithms for Image Interpolation

Numbers of adaptive techniques have been proposed in the literature. In [15, 16, 17], adaptive interpolation is proposed based on logics of local structure of the image and intensity variations that are indistinguishable by human eyes. A nonlinear interpolation algorithm has been proposed in [18, 19] based on a source model and incorporating a local edge fitting technique. A directional image interpolation algorithm based on local analysis of the spatial image structure has been proposed in [20]. Approaches enhancing edges during scaling are proposed in [21, 22].

In [23], a 'directional interpolation' is proposed to overcome common problems associated with interpolation techniques such as image blurring and blocking artifacts resulting in more natural looking magnified images in applications like camcorder zoom. This is accomplished by retaining the sharpness of edges in different arbitrary directions. Local edge information in the image is extracted using DCT coefficients to specify one of five different edge types. The edge type is used in interpolation method to apply in local regions. Then Cubic B-Spline Transform followed by a combination of zero order and Bilinear interpolations are applied. Finally, 5 different Gaussian low-pass filters are applied to reduce the discontinuities and blocking artifacts generated by the interpolation procedure.

In [24], a multi-resolution 'edge-directed covariance based adaptation method' is proposed to statistically model the interpolation to any arbitrary edge orientation so that it does not restrict to specific edge orientations unlike the above in [23]. However, since this technique is computationally quite expensive, it is applied to "edge pixels" only. In other non-edge pixels, the Bilinear interpolation is applied.

The approach proposed in [25] uses local gradient information to adapt the Bilinear and Bicubic interpolation algorithms to yield better perceptual image quality and improved image metrics like the PSNR value. The method also helps to avoid smoothing of edges. In this method, the weights of the input image pixels which in conventional interpolation algorithms are functions of distance only are modified by dividing the normalized local gradients to yield a better interpolation estimation when the gradient changes abruptly in edge regions and across image features.

The technique proposed in [26] produces sharp synthetic and natural images with reduced blurring and undesirable artifacts. First, each pixel to be interpolated is "split" into fourths, which is equivalent to a simple pixel replication. Then a nonlinear correction with a six-pixel subset of a 3×3 mask is used to correct the value of the pixel where the edge is either sharp or smooth. The correction is made by taking appropriate weights of all possible differences of pixels (pairs) and averages of several pixel differences within the sub-mask. With the designation of the interpolator output for all the patterns, the set of weight parameters are obtained using an optimization process to minimize the mean square error of the interpolator output with respect to the desired pixel value.

4. Discrete Wavelet Transform for Image Interpolation

In this section, we propose a novel image interpolation algorithm using the Discrete Wavelet Transform (DWT). The proposed method preserves much of the sharp edge features in the image, and lessens the amount of color artifacts. Effectiveness of the proposed algorithm has been demonstrated based on evaluation of PSNR and ΔE_{ab}^{*} quality metrics with a large database of different types of images. The proposed technique is simple to apply in terms of both software and hardware implementations. First we briefly introduce the concept of DWT in the context of the theme of the current paper and then explain the proposed method for image interpolation.

4.1. Discrete Wavelet Transform

DWT can be implemented by filtering operations with well-defined filter coefficients [5, 6]. In traditional convolution based approach to compute forward DWT, the input signal (x) is filtered separately by a low-pass filter ($\tilde{h}$) and a high-pass filter ($\tilde{g}$). The two output streams are then sub-sampled by simply dropping the alternate output samples in each stream to produce the low-pass (y_L) and high-pass (y_H) subband outputs as shown in Fig. 3. The two filters ($\tilde{h}$, $\tilde{g}$) form the analysis filter bank. The original signal can be reconstructed by a synthesis filter bank (h, g) starting from y_L and y_H as shown in Fig. 3. Given a discrete signal $x(n)$, the output signals $y_L(n)$ and $y_H(n)$ in Fig. 3 can be computed as

$$y_L(n) = \sum_{i=0}^{\tau_L - 1} \tilde{h}(i) \times x(2n - i), \quad y_H(n) = \sum_{i=0}^{\tau_H - 1} \tilde{g}(i) \times x(2n - i)$$

where τ_L and τ_H are the lengths of the low-pass ($\tilde{h}$) and high-pass ($\tilde{g}$) filters respectively. For inverse transform, both y_L and y_H are first up-sampled by inserting zeros in between two samples and then filtered by low-pass (*h*) and high-pass (*g*) filters respectively. Then they are added to obtain the signal (x') as shown in Fig. 3.

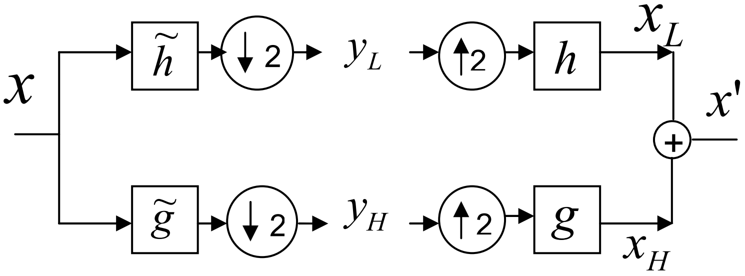
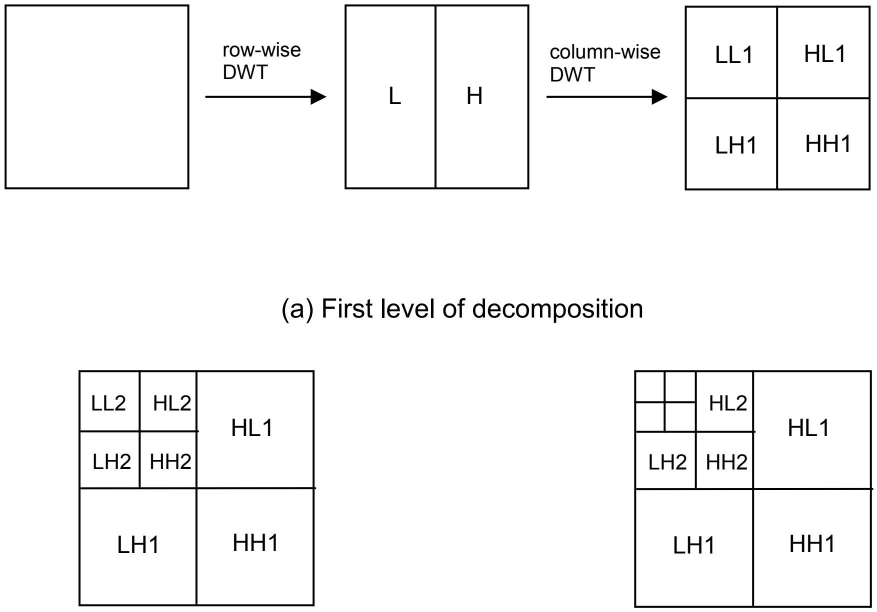


Fig. 3: Analysis and Synthesis filtering in DWT.

For multi-resolution wavelet decomposition, the low-pass subband (y_L) is further decomposed in a similar fashion in order to get the second-level of decomposition, and the process is repeated. The inverse process follows similar multi-level synthesis filtering to reconstruct the signal. Image signals are two-dimensional signals. Since most of the practical two dimensional wavelet filters (e.g. 9/7 bi-orthogonal spline filter) are separable functions, the 2-D DWT can be implemented by applying the 1-D DWT row-wise to produce an intermediate result (L and H subbands in each row) and then applying the same 1-D DWT column-wise on the intermediate result, as shown in Fig. 4(a). In the first level of decomposition, four subbands LL1, LH1, HL1 and HH1 are obtained. Applying the same process in LL1 subband, it produces LL2, LH2, HL2 and HH2 in the second level and so on, as shown in Fig. 4(b)-(c).



(a) First level of decomposition

(b) Second level of decomposition (c) Third level of decomposition

Fig. 4. Three levels of decomposition in 2D DWT.

It is interesting to note that the LL1 subband can be used as a 2:1 down-scaled version of the original image after properly normalizing each sample in the LL subband.

4.2. Image Interpolation by DWT

The DWT based up-sampling technique is explained with an example, as shown in Fig. 5. We up-sample an image (I) of resolution $m \times n$ (say 4×4 as in Fig. 5) to an image (I') of resolution $2m \times 2n$. In the first step, the image is decomposed into four subbands LL , HL , LH and HH by DWT, resulting in the wavelet coefficient image I_{DWT}. The HL and LH subbands contain edge information in horizontal and vertical directions respectively.

The second step is to form a new wavelet coefficient image I'_{DWT} of size $2m \times 2n$. We call it a virtual DWT image, whose LL subband is nothing but the original input image I with each pixel multiplied by a *scaling factor* ***s***. This scale factor ***s*** is set equal to the square of the DC gain of the selected analysis low-pass filter. Depending upon the implementation of the discrete wavelet transform, one may choose a different DC gain for the analysis low-pass filter and Nyquist gain for the analysis high-pass filter, or even choose different filters. For a given n-taps analysis low-pass filer, $\tilde{h}(n)$, the expression $\left|\sum_n \tilde{h}(n)\right|$ denotes the DC gain; while $\left|\sum_n (-1)^n \tilde{g}(n)\right|$ denotes the Nyquist gain of a n-tap high-pass analysis filter, $\tilde{g}(n)$. For more discussion, the readers are referred to the [27].

Hence the dimension of the new LL subband is the same as the resolution $m \times n$ of the original image I. The HH subband of the virtual DWT image (I'_{DWT}) is set all zeros with dimension $m \times n$. The new HL and LH subbands of the virtual DWT image are generated from the original HL and LH subbands (computed in the first step) by inserting zeros in alternate rows and columns as shown in Fig. 5(b).

In the last step, we inverse transform (IDWT) this virtual DWT image (I'_{DWT}) to generate the desired up-sampled image *I'* of resolution $2m \times 2n$.

5. Computational Cost and Timing Analysis

The running time to execute different non-adaptive algorithms is shown in Table II. All the selected non-adaptive methods, namely Bilinear, Bicubic, B-spline, Bell, DWT9/7, Hermite, Lanczos, Mitchell, and Triangle, were implementated using C/C++ programming language. The implementation codes for the 6 filtering-based methods were modified from the source codes provided by [28]. The modifications were mainly to accommodate different image file formats. All the experiments were done on a stand-alone IBM ThinkPad model X32 laptop with Pentium® M, 1.60GHz CPU, and 512MB main memory with 400MHz front-side bus (FSB). The system clock function – “clock( )” in C/C++ programming language was used to record the running time in our implementation.

The computational complexity of both the filtering-based (Triangle, Bell, Hermite, B-Spline, Mitchell, and Lanzos) and DWT-based methods for image interpolation is O(N) and the actual running time depends on the number of filter taps used for interpolation. For example, the Lanczos3 filter uses up to 7 taps in the filtering process which yields the longest running time in all the cases. The Bilinear interpolation considers only 2×2 negihbering pixels and requires least running time. The computation complexity of Bicubic interpolation is O(N^2). The additional computation time for bicubic interpolation is the time to compute the gradients and the cross derivative at each re-sampled pixel location. As we can see from the Table II, the Bicubic requires siganficant amount of running time for each picture in comparison with other methods for the same set of picture in our study. It should be noted that the running time in the Table II is the average running time of up-sampling the red, green, and blue channels.

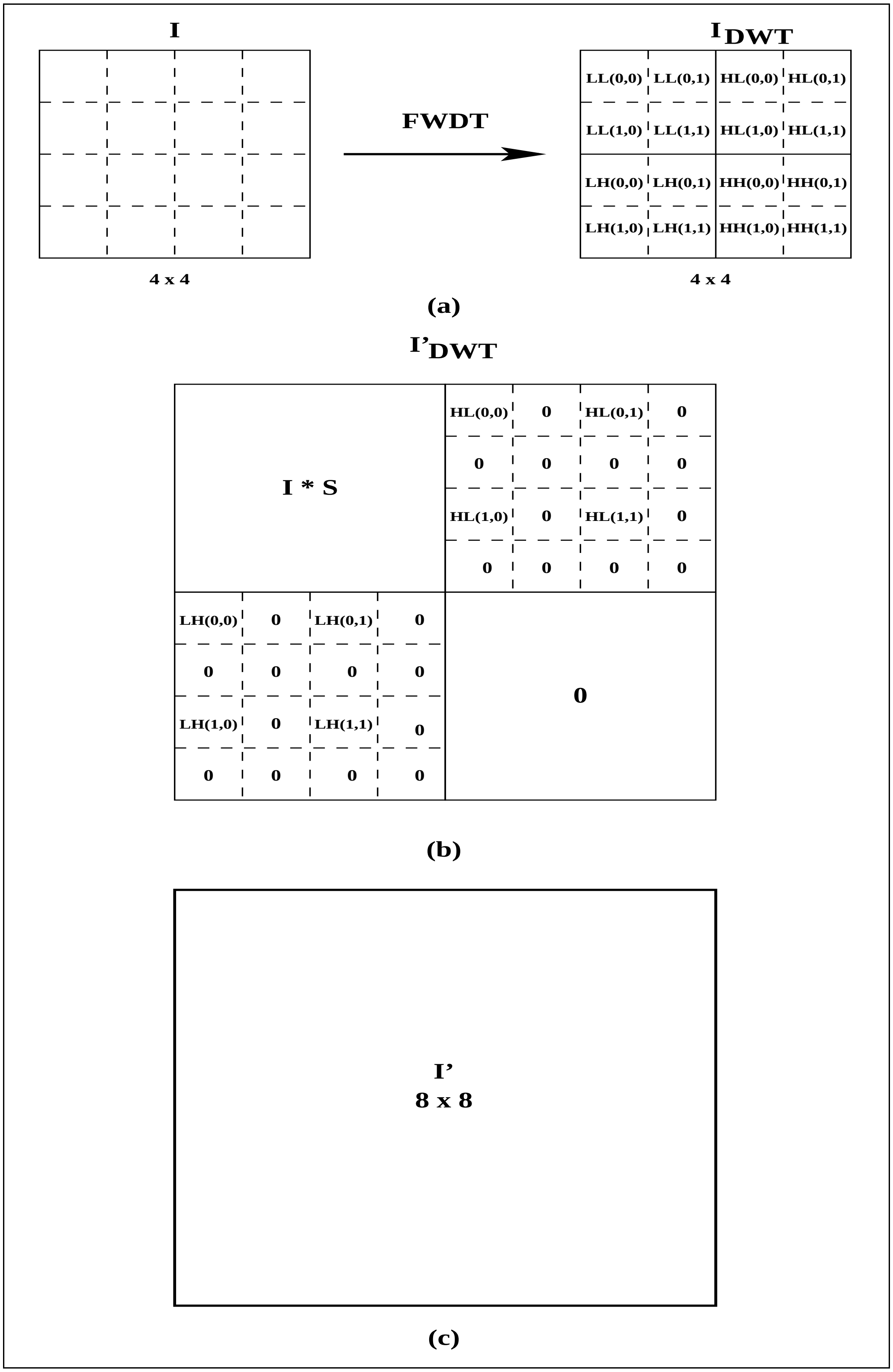


Fig. 5: (a) Forward DWT of input image, (b) construction of the virtual DWT image, and (c) up-scaled image after inverse DWT.

Table II. Running time of different non-adaptive interpolation algorithms.

	Timing (millisecond)							
	1.25M pixels		0.38M pixels			0.44M pixels	0.29M pixels	
	Balloon	Flag	Light-house	Hat	Parrot	Brick	Zebra	Star
Bilinear	131	121	69	68	65	68	54	53
Bicubic	2660	2594	810	820	810	940	640	640
Triangle	386	426	123	125	131	160	119	101
Bell	455	475	150	151	168	172	137	115
B-Spline	519	539	171	168	168	198	133	117
Hermite	380	413	125	137	119	158	109	97
Mitchell	505	540	167	164	155	178	141	115
Lanczos	606	630	195	183	184	207	149	150
DWT9/7	330	450	170	140	150	160	105	100

The adaptive techniques [23, 24, 25, 26] are usually computationally expensive compared to the non-adaptive techniques. The DWT based image interpolation algorithm proposed in this paper is adaptive in nature by virtue of the characteristics of DWT which divides the image into low and high frequency subbands. The directional edge information (vertical, horizontal, and diagonal) in the image are embedded into the high frequency DWT subbands. One can choose wavelet filters based upon characteristics of the image for suitable perceptual quality. However, the underlying operations in the algorithm are not image content dependent and hence non-adaptive in nature.

6. Experiment Results for Image Fidelity

In order to have a quantitative evaluation in our study, we used two commonly used “quality metrics”, peak signal-to-noise ratio (PSNR) and CIELAB ΔE_{ab}^{*} [29]. PSNR depends on another quantity known as the root-mean-squared-error (RMSE). For an image *I’* of size $m \times n$ processed from an input image *I*, the RMSE is given by

$$RMSE = \sqrt{\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}[I(i,j) - I'(i,j)]^2}\ ,$$

where *(i, j)* denotes the position of the pixel in the image. For an 8-bit image, the maximum pixel value is 255 and the peak signal-to-noise ratio in decibel (*dB*) is defined as

$$PSNR = 10\log_{10}\left[\frac{255^2}{RMSE^2}\right].$$

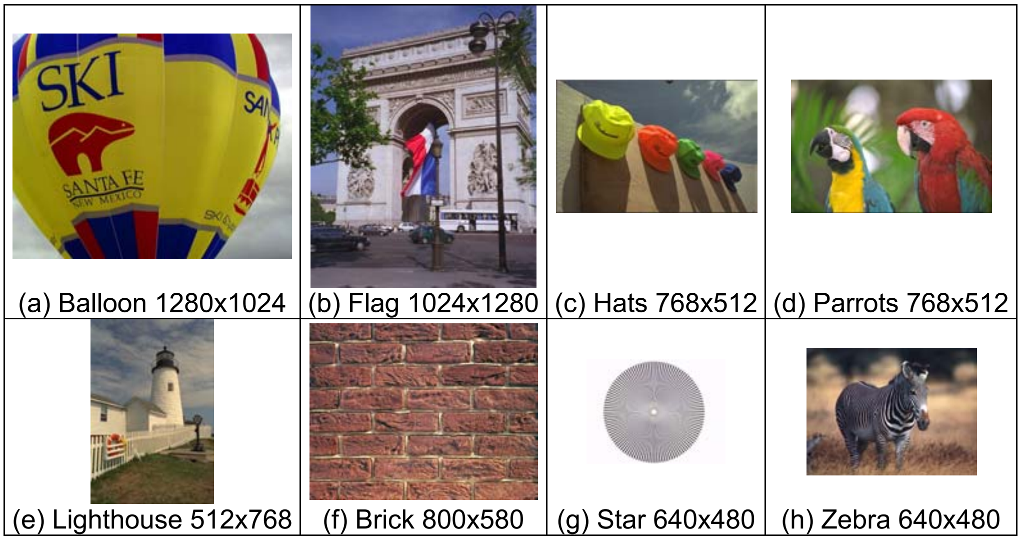
In the context of color imagery, the CIELAB metrics is appropriate for quality measures. The CIELAB, also known as CIE 1976 (L*, a*, b*), color space was developed in 1976 by the Commission Internationale de l’Eclairage (luminosity) or CIE to provide a standard uniform color space. The CIELAB metric is a “perceptual color fidelity” metric and is a measure of how close is the reproduced color (or processed color) to the actual color according to the human eye. The CIELAB ΔE^*_{ab} measures the Euclidean distance between the original and processed image in CIELAB color space. Detail information about how to convert an image from RGB color space to CIELAB color space can be found in [29, 30]. The expression for CIELAB ΔE^*_{ab} is given by

$$\Delta E^*_{ab} = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left\| I(i,j)_{Lab} - I'(i,j)_{Lab} \right\|$$

where $I(i, j)_{Lab}$ and $I'(i, j)_{Lab}$ represent the CIELAB color values of pixel (*i, j*) in the original and processed images, and $\|\cdot\|$ denotes the L2-norm. As discussed in [31], a value of 2.3 on this scale distinguishes the threshold for sensing a difference in color.

The set of images used to report the performance evaluation in this study is shown in Fig. 6. The test set includes two large size images (Balloon and Flag), three natural mid-sized images from Kodak PhotoCD [32] (Hat, Parrots, and Lighthouse), another mid-sized image (Brick) with a lot of edges and texture, and two small-sized images (Star and Zebra) with lots of sharp edges. The test images are first down-sampled by a factor of two in both the vertical and horizontal directions. A simple sub-sampling method is used to down-sample these test images. Hence we intentionally dropped three fourths of the data from the original image. The resultant down-sampled images were then used as inputs for the selected algorithms which up-sampled them to their original size. These up-sampled images are then compared with the original images using the quality metrics discussed early.

Table III shows the average PSNR values over the 3 color channels of each image with respect to different interpolation methods. Table IV shows the ΔE^*_{ab} comparison of test images. As we can see from the PSNR and ΔE^*_{ab} comparison tables, the bilinear and bicubic methods generally perform the worst except for the “Brick” image which is texture type image with a lot of vertical and horizontal edges. The filtering based methods generally yield best performance in term of objective measures. However, it is not necessary true for subjective evaluation based on our observation. The DWT based method is limited to up-sample or down-sample an image by a factor of 2 in both horizontal and vertical directions, but is a good compromise in term of computation without noticeable quality loss.



(a) Balloon 1280x1024 (b) Flag 1024x1280 (c) Hats 768x512 (d) Parrots 768x512

(e) Lighthouse 512x768 (f) Brick 800x580 (g) Star 640x480 (h) Zebra 640x480

Fig. 6. Original test images used in the experiment.

Table III. PSNR comparisons

	PSNR (average over 3 color channels)							
	Balloon	Flag	Light-house	Hat	Parrot	Brick	Zebra	Star
Bilinear	30.1	26.4	25.0	30.1	30.4	**21.7**	22.8	18.0
Bicubic	30.0	26.4	24.8	30.1	30.4	21.5	22.7	18.1
Triangle	34.4	29.7	**27.1**	**31.1**	31.9	19.3	25.3	20.9
Bell	33.5	28.8	26.3	30.7	31.2	20.0	24.3	19.6
B-Spline	32.7	27.7	25.8	30.3	30.6	20.0	23.5	18.7
Hermite	34.4	29.7	**27.1**	**31.1**	31.9	19.3	25.3	20.9
Mitchell	34.7	29.7	26.9	31.0	31.9	19.7	25.3	21.0
Lanczos	**35.3**	**30.1**	26.7	30.8	32.1	19.0	25.6	**21.7**
DWT9/7	33.2	28.0	25.4	30.0	31.0	18.6	24.1	20.2
DWT5/3	33.7	28.1	25.6	30.3	31.0	18.2	24.1	19.7
A-Bilinear (α=0.1)	30.1	26.4	24.8	30.2	30.3	21.6	22.6	17.9
A-Bilinear (α=0.5)	30.0	26.3	24.7	30.2	30.3	21.5	22.5	17.7
A-Bilinear (α=1.0)	30.0	26.3	24.7	30.2	30.3	21.5	22.5	17.6
A-Bicubic (α=0.01)	30.0	26.3	24.7	30.1	30.3	21.5	22.5	17.9
A-Bicubic (α=0.03)	29.8	25.9	24.4	30.0	30.0	21.1	21.9	17.1
A-Bicubic (α=0.05)	29.5	25.5	24.0	29.8	29.6	20.8	21.4	16.4
2X Int.	29.9	24.8	23.6	28.9	28.8	20.0	21.1	15.5
Edge-Directed	34.6	29.4	26.4	**31.1**	**32.2**	19.3	**26.1**	21.5

Table IV. ΔE^*_{ab} comparisons

	ΔE^*_{ab}							
	Balloon	Flag	Light-house	Hat	Parrot	Brick	Zebra	Star
Bilinear	2.69	2.31	2.57	1.62	1.55	5.89	3.32	4.67
Bicubic	2.72	2.32	2.63	1.64	1.59	5.87	3.44	4.51
Triangle	1.68	**1.44**	1.73	**1.15**	**0.99**	7.04	2.12	2.87
Bell	2.21	1.93	2.20	1.43	1.25	6.73	2.93	4.25
B-Spline	2.41	2.14	2.37	1.54	1.36	6.79	3.24	4.80
Hermite	1.68	**1.44**	1.73	**1.15**	**0.99**	7.04	2.12	2.87
Mitchell	1.77	1.59	1.95	1.27	1.09	6.83	2.43	3.35
Lanczos	**1.66**	1.45	1.90	1.26	1.08	7.40	2.34	2.78
DWT9/7	2.27	2.07	2.52	1.62	1.42	7.92	3.38	4.17
DWT5/3	2.03	1.86	2.18	1.44	1.24	8.14	2.86	3.81
A-Bilinear (α=0.1)	2.68	2.31	2.57	1.63	1.57	**5.59**	3.29	4.58
A-Bilinear (α=0.5)	2.71	2.34	2.60	1.64	1.59	5.65	3.32	4.61
A-Bilinear (α=1.0)	2.73	2.34	2.60	1.65	1.60	5.66	3.33	4.63
A-Bicubic (α=0.01)	2.74	2.36	2.68	1.67	1.61	5.63	3.76	5.03
A-Bicubic (α=0.03)	2.83	2.49	2.80	1.71	1.67	6.05	4.20	6.39
A-Bicubic (α=0.05)	2.92	2.64	2.92	1.76	1.74	6.51	4.57	7.61
2X Int.	3.18	2.81	3.05	1.91	1.81	7.27	4.12	8.04
Edge-Directed	1.79	1.55	1.91	1.23	1.07	7.07	**2.03**	**2.66**

7. Conclusion

Image processing technologies play central role to process visual information in order to make it suitable for multimedia applications and visual perception. In this era of Internet and multimedia communication, sizing and resizing of images, to make it suitable for viewing, transmission, downloading, sharing, editing, and further processing, became very vital. Image interpolation became a versatile and widely used tool in image processing today because of its numerous applications in diverse areas ranging from computer graphics, rendering, editing, medical image reconstruction, to online image viewing to name a few. Basically, image interpolation algorithms convert or resize a digital image from one resolution to another resolution without loosing the visual content in the picture. Image interpolation algorithms are grouped into two categories – non-adaptive and adaptive. In this paper, we discussed some key non-adaptive and adaptive algorithms for image interpolation and presented experimental results in terms of computation requirements and image fidelity. We proposed a new algorithm for image interpolation based on popular discrete wavelet transform. Computation of the algorithm is non-adaptive in nature, but the inherent features of DWT give it a flavor of adaptive algorithm also. We have compared the experimental result of this algorithm with other algorithms in the literature.

References

[1] T. Acharya and A. K. Ray, Image Processing: Principles and Applications, John Wiley & Sons, Inc., Hoboken, NJ, 2005.

[2] Adobe® CS2 software. (http://www.adobe.com/products/photoshop/)

[3] IrfanView graphic viewer. (http://www.irfanview.com/)

[4] T. Acharya, P. -S. Tsai, "Image up-sampling using Discrete Wavelet Transform," in Proceedings of the 7th *International Conference on Computer Vision, Pattern Recognition and Image Processing* (CVPRIP 2006), in conjunction with 9th *Joint Conference on Information Sciences* (JCIS 2006), October 8-11, 2006, Kaohsiung, Taiwan, ROC, pp. 1078-1081.

[5] S. Mallat, A Theory for Multi-resolution Signal Decomposition: The Wavelet Representation, IEEE Trans. on Pattern Analysis and Machine Intelligence, 11 (7) (1989) 674-693.

[6] T. Acharya and P.-S. Tsai, JPEG2000 Standard for Image Compression: Concepts, Algorithms, and VLSI Architectures, John Wiley & Sons, Inc., NJ, 2005.

[7] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, Numerical Recipes in C++: The Art of Scientific Computing, 2nd ed., Cambridge University Press, New York, NY, 2002.

[8] J. A. Parker, R. V. Kenyon and D. E. Troxel, "Comparison of Interpolating Methods for Image Resampling," *IEEE Trans. on Medical Imaging*, vol. MI-2, pp. 31-39, 1983.

[9] D. Kirk, *Graphics Gems III (IBM Version)*, Academic Press, San Diego, CA, 1992.

[10] P. Thevenaz, T. Blu and M. Unser, "Image Interpolation and Resampling," *Handbook of Medical Imaging, Processing and Analysis*, pp. 393-420, 2000.

[11] T. Lehmann, C. Gonner, and K. Spetzer, "Survey: Interpolation Methods in Medical Image Processing," *IEEE Trans. Med. Imaging*, 18:1049-1067, 1999.

[12] D.P. Mitchell, and A.N. Netravali, "Reconstruction Filters in Computer Graphics," *ACM Siggraph Computer Graphics*, Vol. 22, No. 4, pp. 221-228, 1988.

[13] H. S. Hou and H. C. Andrews, "Cubic Splines for Image Interpolation and Digital Filtering," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-26, pp. 508-517, 1978.

[14] K. Turkowski, "Filters for Common Resampling Tasks," In A. S. Glassner, editor, *Graphics Gems I*, Academic Press, pp. 147-165, 1990.

[15] Y. Wang, and S. Mitra, "Edge Preserved Image Zooming," *Proc. of European Signal Process*, *EURASIP-88*, pp. 1445-1448, Grenoble, France, 1988.

[16] Y. Wang, and S. Mitra, "Motion/Pattern Adaptive Interpolation of Interlaced Video Sequences," *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, *ICASSP-91*, vol. 4, pp. 2829-2832, Toronto, Ont., Canada, April 1991.

[17] S. Thurnhofer, M. Lightstone, and S. Mitra, "Adaptive Interpolation of Images with Application to Interlaced-to-Progressive Conversion," *Proc. SPIE-int. Soc. Opt. Eng.*, vol. 2094, pp. 614-625, 1993.

[18] K. Jensen, and D. Anastassiou, "Spatial Resolution Enhancement of Images Using Nonlinear interpolation," *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, *ICASSP-90*, pp. 2045-2048, Albuquerque, NM, 1990.

[19] K. Jensen, and D. Anastassiou, "Subpixel Edge Localization and Interpolation of Still Image," *IEEE Trans. Image Processing*, 4(3), pp. 285-295, 1995.

[20] V. Algazi, G. Ford, and R. Potharlanka, "Directional Interpolation of Images Based on Visual Properties and Rand Order Filtering," *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, *ICASSP-91*, Vol. M12.25, Toronto, Canada, 1991.

[21] W. Bender, and C. Rosenberg, "Image Enhancement Using Non-uniform Sampling," *Proc. SPIE-int. Soc. Opt. Eng.*, vol. 1460, pp. 59-70, 1991.

[22] S. Thurnhofer, and S. Mitra, "Edge-Enhanced Image Zooming," *Optical Eng.*, 35(7), pp. 1862-1869, 1996.

[23] K. P. Hong, J. K. Paik, H. J. Kim, and C. H. Lee, "An Edge-Preserving Image Interpolation System for a Digital Camcorder," *IEEE Trans. on Consumer Electronics*, vol. 42, no. 3, August 1996.

[24] X. Li and M. T. Orchard, "New Edge-Directed Interpolation," *IEEE Trans. on Image Processing*, vol. 10, no. 10, October 2001.

[25] J.W. Hwang and H.S. Lee, "Adaptive Image Interpolation Based on Local Gradient Features," *IEEE Signal Processing Letters*, vol.11, no.3, March 2004.

[26] S. Carrato and L. Tenze, "A High Quality 2× Image Interpolator," *IEEE Signal Processing Letter*, vol. 7, no. 6, June 2000.

[27] M. Rabbani and D. Santa-Cruz, "The JPEG 2000 still-image compression standard," course note given at the 2001 International Conference in Image Processing (ICIP), Thessaloniki, Greece, October 11, 2001. (http://jj2000.epfl.ch/jj_publications/index.html)

[28] http://www.acm.org/tog/GraphicsGems/gemsiii/filter_rcg.c

[29] M.D. Fairchild, Color Appearance Models, Reading, MA: Addison Wesley, 1997.

[30] Recommendations on Uniform Color Spaces, Color Difference Equations, Psychometric Color Terms, C.I.E, Supplement no. 2 to CIE publication no. 15(E.-I 31) 1971/(TC-1.3), 1978.

[31] G. Sharma and H.J. Trussell, Digital Color Imaging, IEEE Transactions on Image Processing, 6 (7) (1997) 901-932.

[32] http://r0k.us/graphics/kodak/.

Dr. Tinku Acharya is the Chief Technology Officer and a co-founder of Avisere Inc., Arizona, and the Managing Director of its subsidiary in Kolkata, India. He is an Adjunct Professor in the Department of Electrical Engineering, Arizona State University. He received his B.Sc. (Honors) in Physics, B.Tech and M.Tech in Computer Science from the University of Calcutta, India and Ph.D. in Computer Science from the University of Central Florida, Orlando, USA.

Dr. Acharya leads development of emerging '*Intelligent Video Analytics'* technology to automate surveillance systems in IP digital video networks. Before founding Avisere, he served in Intel Corporation (1996-2002) to lead several R&D teams to develop algorithms and architectures for image and video processing, PC-based digital camera, reprographics architecture for color photocopiers, multimedia architecture for 3G cellular telephony, analysis of advanced microprocessor, etc. He developed the `*image processing chain*' to pragmatically map into silicon for the first low-cost dual-mode digital camera (Intel's WebCam). His works became foundation of the MXP5800/5400 media processor from Intel, capable of processing 10 BOPS and enabled a set of Internet capable photocopiers from Xerox and Fuji/Xerox. Before Intel, he was a consulting engineer at AT&T Bell Lab (1995-1996), a faculty in Institute of Advanced Computer Studies, University of Maryland at College Park (1994-1995). He served in National Informatics Center, Government of India (1988-1990) and also as a visiting faculty at Indian Institute of Technology (IIT), Kharagpur. He collaborated with Palo Alto Research Center (PARC) in Xerox Corporation, Eastman Kodak Corporation, and many other institutions worldwide.

Dr. Acharya is inventor of 113 US patents and 14 European patents. He contributed to over 90 papers published in international journals, conferences, and book chapters. He is author of four books - (i) *Image Processing: Principles and Applications* (Wiley, 2005), (ii) *JPEG2000 Standard for Image Compression: Concepts, Algorithms, and VLSI Architectures* (Wiley, 2004), (iii) *Information Technology: Principles and Applications* (Prentice-Hall India, 2004), and (iv) *Data Mining: Multimedia, Soft Computing and Bioinformatics* (Wiley, 2003). He was recognized the *Most Prolific Inventor* in Intel Corporation Worldwide in 1999 & 2001, and Intel Arizona site for five consecutive years (1997-2001). He supervised two doctoral dissertations.

Dr. Acharya is Fellow of *Indian National Academy of Engineering*, *Institution of Electronics and Telecommunication Engineers*, *Institute of Engineers*, Senior Member of IEEE, and Member of ACM. He served in US National Body of the JPEG2000 standards committee. His research interests include computer vision, VLSI signal processing, multimedia computing, and multimedia data mining.

Dr. Ping-Sing Tsai received his BS degree in information and computer engineering from the Chung-Yuan Christian University in 1985 and his PhD degree in computer science from the University of Central Florida in 1995. Dr. Tsai was with Intel Corporation, Chandler, Arizona from 1997 – 2002 as a Senior Systems R&D engineer. Since 2003, he is an Assistant Professor in the Department of Computer Science, the University of Texas – Pan American. His research interests include computer vision, pattern recognition, image processing, and multimedia computing. Dr. Tsai is a senior member of IEEE and ACM.

E-mail: *tinku.acharya@ieee.org*, *pstsai@ieee.org*