PAPER

Shape-Direction-Adaptive Lifting-Based Discrete Wavelet Transform for Arbitrarily Shaped Segments in Image Compression

Sheng-Fuu LIN[†a)], *Nonmember and* **Chien-Kun SU**[†,††b)], *Student Member*

SUMMARY In this paper, a new lifting-based shape-direction-adaptive discrete wavelet transform (SDA-DWT) which can be used for arbitrarily shaped segments is proposed. The SDA-DWT contains three major techniques: the lifting-based DWT, the adaptive directional technique, and the concept of object-based compression in MPEG-4. With SDA-DWT, the number of transformed coefficients is equal to the number of pixels in the arbitrarily shaped segment image, and the spatial correlation across subbands is well preserved. SDA-DWT also can locally adapt its filtering directions according to the texture orientations to improve energy compaction for images containing non-horizontal or non-vertical edge textures. SDA-DWT can be applied to any application that is wavelet based and the lifting technique provides much flexibility for hardware implementation. Experimental results show that, for still object images with rich orientation textures, SDA-DWT outperforms SA-DWT up to 5.88 dB in PSNR under 2.15-bpp (bit / object pixel) condition, and reduces the bit-budget up to 28.5% for lossless compression. SDA-DWT also outperforms DA-DWT up to 5.44 dB in PSNR under 3.28-bpp condition, and reduces the bit-budget up to 14.0%.

key words: *compression, textures, set-partitioning embedded block coder (SPECK), object-based video coding, shape-direction-adaptive DWT (SDA-DWT)*

1. Introduction

The conventional separable 2-D DWT can be implemented by consecutively applying the 1-D DWT in horizontal and vertical directions, or vice versa. Hence, only these two directions of the high-pass filters have vanishing moments. For images containing large amount of edges which are not vertical or horizontal, the conventional DWT cannot provide efficient representations, since there are many large amplitude coefficients in the high frequency subbands of the transformed images. In order to solve this problem, we need to design a DWT whose directions are not fixed to vertical or horizontal. On the contrary, the new separable 2-D DWT will be capable of choosing the best directions for executing two 1-D DWTs. Such DWTs that can choose the optimal transform directions are usually called direction-adaptive DWTs. Recently, Ding *et al.* proposed the adaptive directional lifting-based wavelet transform (ADL-DWT) [1] for image compression, and they claimed that ADL-DWT can improve the compression performance of a texture-rich image up to 2.0 dB. In [1], they used the lifting scheme [3], [4] and the sinc-interpolation [5] to design ADL-DWT. About the same time, Chang *et al.* also proposed a direction-adaptive discrete wavelet transform (DA-DWT) [6] for image compression which was based on the lifting scheme, too. DA-DWT does not involve the sub-pixel interpolation of ADL-DWT, but they use the existing input samples for prediction and update. DA-DWT can attain a gain up to 2.5 dB in PSNR over the traditional DWT for typical testing images. From [1] and [6], we clearly know that DWT with direction-adaptive capability can improve the performance of the conventional DWT to a new level, but the extra cost is complicated computation and the side information processing, which contains the direction information, to be stored, coded, and transmitted. Besides the complexity and side information, ADL-DWT and DA-DWT are designed for compressing a rectangular image, and they cannot be used to object-based or arbitrary-shape image compression directly.

Both ADL-DWT and DA-DWT are based on the lifting-based DWT, since the lifting-based DWT is convenient for direction-adaptive functionality realization and hardware implementation. The convolution-based DWT or FIR (finite impulse response) bank structure DWT proposed by Mallat [7] is the traditional method to implement DWT. The convolution-based DWT suffers from the problems that it is complex and difficult for hardware implementation. Therefore, Daubechies and Sweldom proposed the lifting-based DWT [3] which is less complicated than the convolution-based DWT. The main concept of the lifting-based DWT is to break-up the low-pass and high-pass wavelet filters into a sequence of lower-triangular and upper-triangular matrices, and implement the filter by banded matrix multiplications. Because of low complexity, ease for hardware implementation, and the capability of lossless reconstruction, the lifting-based DWT was recommended by JPEG2000.

An important feature of MPEG-4 is the functionality that the compressed forms of visual objects are available, and this feature provides great flexibility for manipulating visual objects in multimedia applications. For this functionality of MPEG-4, many coding techniques for coding arbitrarily shaped visual object were developed, and the shape-adaptive discrete cosine transform (SA-DCT) [8] is the most

Manuscript received February 22, 2008.
Manuscript revised May 19, 2008.

[†]The authors are with the Department of Electrical and Control Engineering of National Chiao Tung University, Hsinchu, Taiwan.
[††]The author is also with the Department of Electrical Engineering of Chung Hua University, Hsinchu, Taiwan.
a) E-mail: sflin@mail.nctu.edu.tw
b) E-mail: cks@chu.edu.tw

DOI: 10.1093/ietisy/e91–d.10.2467

popular one to code the texture of the intra frame of a visual object in video coding. Since SA-DCT divides the object to be compressed into many 8-by-8 blocks, the non-vertical and non-horizontal boundaries of objects can not be represented perfectly, i.e. some positions of the 8-by-8 boundary block are not in the object. In order to overcome the problems of SA-DCT, Li *et al.* proposed the shape-adaptive discrete wavelet transform (SA-DWT) [9] for coding the texture of an arbitrarily shaped visual object. SA-DWT uses DWT to replace DCT, and it can handle arbitrarily shaped 2-D objects and gets better performance by using the more complicated algorithm. Lu and Pearlman combined the shape-adaptive DWT and the set-partitioning embedded block coder (SPECK) [10] to propose the object-based SPECK algorithm [11] for coding the arbitrarily shaped objects of the intra frames in MPEG-4.

In this paper, we propose a shape-direction-adaptive lifting-based DWT (SDA-DWT). By using SDA-DWT, we can perform directional adaptive DWT on an arbitrarily shaped and partitioned visual object while the shape mask and partition of the object image are given. SDA-DWT can be directly applied to still image compression and object-based visual compression in MPEG-4 with high efficiency. Experimental results show that SDA-DWT outperforms SA-DWT (a conventional-separable-2D-DWT-based method) by 5.76 dB in PSNR for the test image under 1-bpp condition, and reduces 28.5% bit-budget of the coded bit-stream. For our test image, SDA-DWT also outperforms DA-DWT (a lifting-based method) by 4.88 dB under 1-bpp condition, and reduces the bit-budget of the coded bit-stream 14.0%. The remainder of this paper proceeds as follows. In Sect. 2, the proposed SDA-DWT is described in detail. The lifting-based DWT, the adaptive directional DWT, and the shape-adaptive DWT are also included in this section. Experimental results are given in Sect. 3, followed by the conclusions in Sect. 4.

2. Shape-Direction-Adaptive Discrete Wavelet Transform (SDA-DWT)

In this section, we present a new discrete wavelet transform, which is both shape-adaptive and direction-adaptive and named shape-direction-adaptive discrete wavelet transform (SDA-DWT). Besides the direction-adaptive capability like ADL-DWT or DA-DWT, SDA-DWT is capable of handling arbitrarily shaped segments. Some related topics such as the lifting structure of DWT, the direction-adaptive DWT, and the shape-adaptive DWT are introduced in this section, too.

2.1 Lifting-Based Structure

Wavelet transform is well known as a multi-resolution analysis that provides many advantages: joint space-spatial frequency localization, clustered wavelet coefficients of significance with strong correlations between subbands, and exact reconstruction, which are truly beneficial to image compression. Discrete wavelet transform (DWT) decomposes a signal: $S_l(n)$ at resolution l into two components:

$$S_{l+1}(n) = \sum_k S_l(k)h(2n-k), \tag{1}$$

$$D_{l+1}(n) = \sum_k S_l(k)g(2n-k), \tag{2}$$

where $S_l(n)$ is its approximation at the next coarser resolution $l+1$, $D_l(n)$ is the detail information between the two successive resolutions: l and $l+1$, $h(n) = \langle \phi, \phi_{-1,-n} \rangle$, $g(n) = \langle \psi, \phi_{-1,-n} \rangle$, $\langle \cdot, \cdot \rangle$ is an inner product operator, ψ is a valid (mother) wavelet, ϕ is the scaling function, and $\phi_{-1,-n}(x) = 2^{-1/2}\phi(x/2-n)$. The original signal can be exactly reconstructed from $S_{l+1}(n)$ and $D_{l+1}(n)$ by using the following inverse DWT (IDWT):

$$S_l(n) = \sum_k S_{l+1}(k)\tilde{h}(n-2k) + \sum_k D_{l+1}(k)\tilde{g}(n-2k), \tag{3}$$

where $\tilde{h}(n) = h(-n)$ and $\tilde{g}(n) = g(-n)$. The DWT whose transform is based on Eqs. (1) and (2) and inverse transform on Eq. (3), is called the convolution-based DWT. Figure 1 shows the block diagrams of the convolution-based one-level DWT and IDWT, where s and d are equivalent to S_{l+1} and D_{l+1}, respectively, and x is equivalent to S_l. The convolution-based DWT was widely used for researching and implementing DWT for a long time, so most researchers are familiar to it. The disadvantages of convolution-based DWT are its complexity, large storage space requirement, and difficulty of hardware implementation. Daubechies and Swelden had proposed a new approach, called lifting-based DWT, for implementing DWT. The lifting-based scheme is to decompose a discrete wavelet transform into a finite sequence of simple filtering steps, which are called lifting steps. Using the language of algebraists, the decomposition of lifting-based DWT corresponds to a factorization of the polyphase matrix of the wavelet into elementary matrices. The lifting-based approach can provide advantages such as in-place implementation of the fast DWT, capability of integer-to-integer transform, ease for hardware implementation, less storage space requirement, and flexibility for some adaptations on DWT.

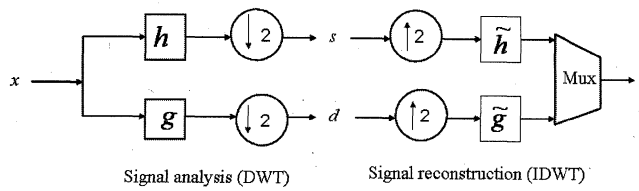


Fig. 1 The block diagrams of the 1-level 1-D convolution-based DWT and IDWT.

For the lifting structure, each finite impulse response (FIR) wavelet filter is factored into several pairs of lifting steps. One pair of lifting steps includes a prediction step

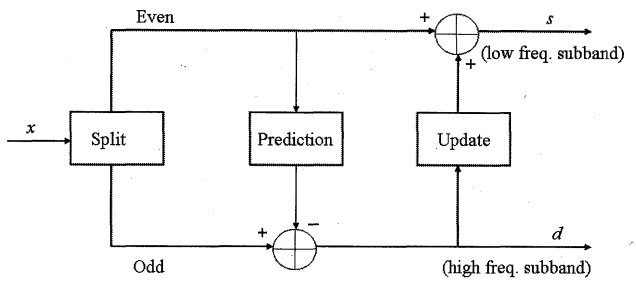


Fig. 2 The block diagram of the lifting-based DWT containing one pair of lifting steps.

followed by one update step. In this paper, we only discuss the 5/3 wavelet, and the 5/3 wavelet can be realized with only one pair of lifting steps. The block diagram in Fig. 2 shows the structure of the lifting-based DWT containing one prediction step and one update step. For the one-dimension lifting-based DWT, the input samples are classified into two categories (even and odd) first. Then, each odd sample is predicted by some specific even-neighbor samples (This depends on the wavelet type.), and replaced by the residual obtained from subtracting the odd sample by the prediction value. After the prediction step, in the followed update step, each of the even samples is updated by the value generated from its odd-neighbor samples. Note that, in the update step, the odd samples are not the original inputs, since they have been changed in the preceding prediction step. Finally, the outputs are down-sampled to produce the low frequency subband and the high frequency subband and complete a lifting-based DWT with one pair of lifting steps. The lifting-based IDWT can be implemented by reversing the steps in the corresponding lifting-based DWT. The following equations are a realization of the 5/3 wavelet lifting-based DWT:

$$y(2k+1) = x_{ext}(2k+1) - [x_{ext}(2k) + x_{ext}(2k+2)]/2, \quad (4)$$

$$y(2k) = x_{ext}(2k) + [y(2k-1) + y(2k+1) + 2]/4, \quad (5)$$

$$d(k) = y(2k+1), \quad (6)$$

$$s(k) = y(2k). \quad (7)$$

For the 5/3 wavelet used in this paper, Eqs. (4) and (5) are the prediction function and the update function, respectively. Equations (6) and (7) are two down-sampling relations used to generate the low frequency subband output $s(k)$ and the high frequency subband output $d(k)$, respectively, where x is the 1-D input data, and x_{ext} means the symmetric extended version of x. Assume that $x = \{1, 2, 3, 4, 5\}$, and we have $x_{ext} = \{\cdots, 4, 3, 2, 1, 2, 3, 4, 5, 4, 3, 2, \cdots\}$. Assuming the length of x is even, $0 \le k \le$ (length of x)/2. The operations corresponding to Eqs. (4) and (5) can be also represented as Fig. 3. The lifting-based IDWT of the 5/3 wavelet is shown in Eqs. (8)-(11).

$$y(2k+1) = d(k), \quad (8)$$

$$y(2k) = s(k). \quad (9)$$

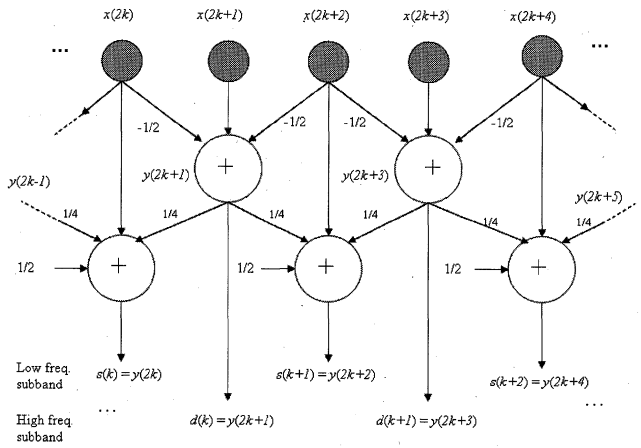


Fig. 3 The structure of a lifting-based one-dimensional 5/3-wavelet DWT.

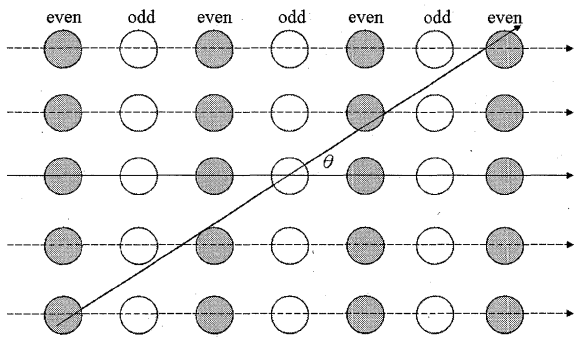


Fig. 4 A direction selection example with angle θ in 1-D 'horizontal' DWT.

$$x'(2k) = y_{ext}(2k) - [y_{ext}(2k-1) + y_{ext}(2k+1) + 2]/4, \quad (10)$$

$$x'(2k+1) = y_{ext}(2k+1) + [x'(2k) + x'(2k+2)]/2, \quad (11)$$

where y_{ext} is the symmetric extended version of y and x' is the reconstruction version of the input x. The lifting-based IDWT begins from applying Eqs. (8) and (9) for up-sampling d and s to produce y. Then, the even input samples are reconstructed by using Eq. (10). Finally, using Eq. (11) to reconstruct the odd samples of x.

2.2 Directional Lifting DWT

The conventional separable 2-D DWT can be implemented by consecutively applying 1-D DWT in horizontal and vertical directions, or vice versa. That means if we use the lifting structure to implement 2-D DWT, the prediction and update directions are parallel to the horizontal or vertical axes. For the lifting-based DWT whose directions of prediction and update steps are adaptive is called a direction-adaptive DWT (DA-DWT) or adaptive directional lifting-based DWT (ADL-DWT). Figure 4 shows a direction selection in the lifting step of a direction-adaptive DWT, where

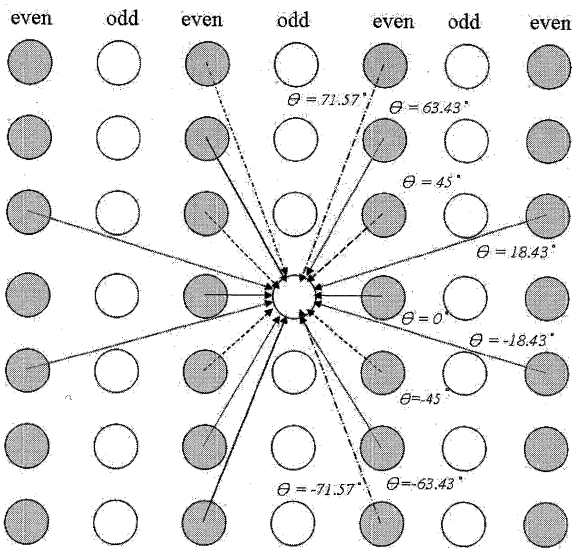


Fig. 5 Nine prediction directions for an odd sample in 1-D 'horizontal' DWT.

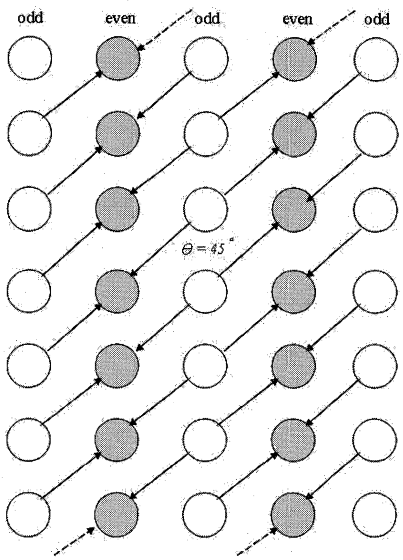


Fig. 6 The update stage with $\theta = 45^o$.

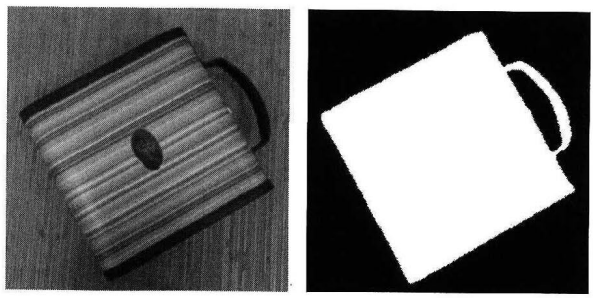
the direction line intersects the horizontal axis by an angle θ. Applying sub-pixel technique, although the angle θ can be any value between 0 and $\pi/2$ (radians), only nine directions were used in [1]. In [6], they also used nine directions for prediction and update, but these nine directions were not the same as those in [1]. In this paper, we use the directions in [6], because this method does not involve complex sub-pixel computation and has better performance. Figure 5 shows the nine directions and their corresponding neighbors of an odd sample in prediction step of a 5/3-wavelet DWT. In Fig. 6, each of the even samples is updated by its two odd neighbors along the line with $\theta = 45^o$.

ADL-DWT [1] and DA-DWT [6] are proposed to compress a rectangular image by dividing the whole image into a lot of fixed-size small square blocks. After dividing an image into many small square blocks, the optimal direction for directional lifting DWT of each block is determined. Then, some connected blocks with the same lifting direction are grouped to form a large rectangular block with the same direction for saving the bits of side information [1]. Different from the method in [1], the method used in [6] splits a larger square block into several small rectangular blocks instead of merging some small square blocks to form a larger rectangular block. For lossless image compression, the best direction (θ) of prediction and update of the directional lifting DWT is the direction that spends the least amount of bits to compress this (square or rectangular) block. For lossy image compression, the best direction of the directional lifting DWT in a block should be the one that has the highest PSNR value for a given bit-budget. Either lossless or lossy image compression, the best direction selection should have strong energy compaction effect in the low frequency subband. On the other hand, energy compaction in low frequency subband is equivalent to that the energy left in high frequency subband is little. Hence, the optimal direction selection can be approximately determined by choosing the direction in which the directional DWT has the smallest absolute sum of the coefficients in the high frequency subband.

The last step of the 1-D directional DWT is a subsampling stage, and the subsampling method is just like the way in the conventional DWT, i.e. the subsampling direction is the horizontal direction for a 'row' directional adaptive DWT. After the 'row' directional adaptive DWT is complete, the 'column' direction DWT is performed on the whole segment block, and the last step is the subsampling step following the 'column' directional DWT in a one-level direction-adaptive DWT. The realization of the 'column' direction-adaptive DWT is the same as the 'row' direction-adaptive DWT, if the segment block after 'row' direction-adaptive DWT is rotated clockwise 90 degrees.

2.3 DWT for Arbitrarily Shaped Segments

Because of fast growth of multimedia applications, the needs of searching, accessing, indexing, and manipulating visual information at the semantically meaningful object level are becoming more and more urgent. MPEG-4 standard supports such a functionality of making a visual object available in the compressed form, and this functionality provides flexibility for manipulating a visual object in multimedia applications and improves the compression efficiency in very low bit-rate coding. There are two major parts in an object-based video coding. One is the intra frame coding, and the other is the inter frame coding. The inter frame coding involving motion prediction, which will not be discussed here, and we focus on the intra frame coding in this paper. The intra frame coding of the object-based video coding can be divided into object shape coding and object texture coding. The shape mask (Fig. 7 (b)) is used to represent the region that the object occupied, and the simplest shape mask can be a binary figure which has value 1 for each pixel in the object and value 0 for each pixel outside the object. Thus,



(a) (b)

Fig. 7 Test object1 and its shape mask: (a) 256 × 256 object 1 image with background, (b) shape mask of object 1.

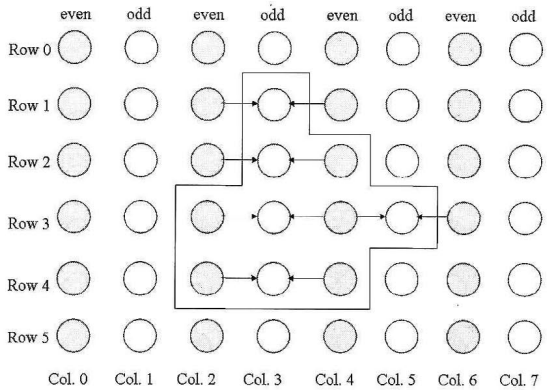


Fig. 8 An arbitrarily shaped segment and the relation of its even and odd pixels in the prediction step of the 1-D row direction lifting-based DWT.

using the shape mask, the object in an image can be easily segmented. The most popular technique for object texture coding is the shape-adaptive DCT (SA-DCT) [8], which uses 8-by-8 blocks to approximate the shape of the object to be coded. Since an object usually can not be represented by 8-by-8 blocks perfectly, a lot of boundary blocks do not totally reside in the object and make this method inefficient. S. Li *et al.* proposed a shape-adaptive discrete wavelet transform (SA-DWT) for arbitrarily shaped visual object coding [9], and they used SA-DWT for the texture coding of the intra frame part in object based video coding. Lu *et al.* also proposed an object texture coding technique [11] that combined SA-DWT and SPECK algorithm [10]. The experimental results in [11] showed that SA-DWT with extensions of zerotree entropy coding (ZTE) outperforms SA-DCT up to 0.97 dB in Y-plane PSNR, 1.29 dB in U-plane PSNR, and 0.89 dB in V-plane PSNR, for the Akiyo sequence (CIF) at 1.0 bpp.

The works of [9] and [11] used the convolution-based DWT, and both of them involved complicated computation. Using the lifting-based DWT and global even-odd relation we can realize SA-DWT easily. Figure 8 shows that an arbitrarily shaped segment contains 10 pixels in a 6-by-8-pixel image, and it also shows the relation of even and odd pixels in the prediction step. In Fig. 8, the two arrows, pointing to each odd pixel, are used to indicate that the odd pixel's two nearest neighbors in the same row are used to predict

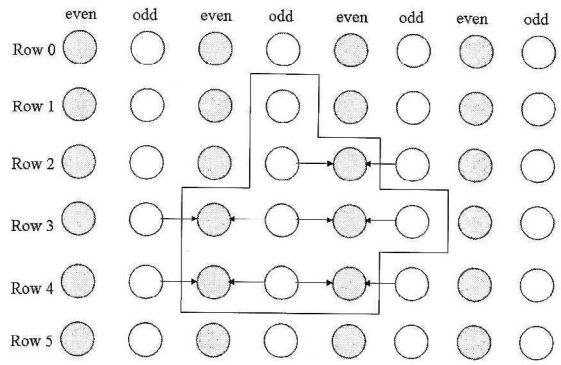


Fig. 9 An arbitrarily shaped segment and the relation of its even and odd pixels in the update step of the 1-D horizontal lifting-based DWT.

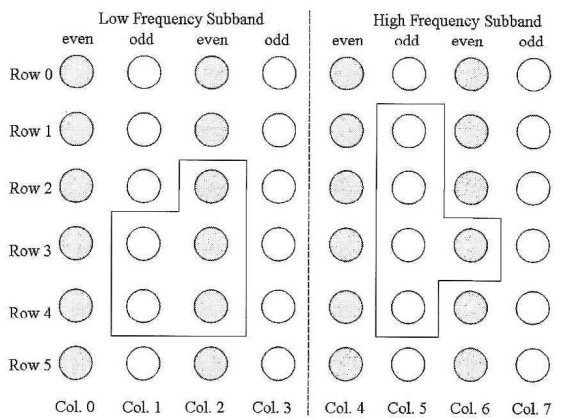


Fig. 10 The subsampling result of the arbitrarily shaped segment in Figs. 8 and 9 after 1-D horizontal lifting-based DWT.

the odd pixel. For the 5/3 wavelet DWT, the prediction value of each odd pixel in the lifting-based horizontal 1-D DWT is the mean value of its right and left neighbors. After the prediction step, the residual that each odd pixel subtracts its prediction value is stored in the position of the odd pixel. If the even neighbor does not in the segment, the symmetric extension is used to generate a new even pixel value for prediction. For the single point in a row (e.g. the pixel at row 1 and column 3), its two neighbors for prediction are set to zero. Figure 9 shows the update relation of the arbitrarily shaped segment when the lifting-based horizontal 1-D 5/3 DWT is performed on the segment. Each even pixel in the segment is updated by using Eq. (5) in the update step, and the corresponding pixels (coefficients) are its left and right odd neighbors in the 1-D lifting-based horizontal 5/3-wavelet DWT. The processing methods of the symmetric extension and single points are the same as those in the prediction step. The last step of the 1-D lifting-based DWT is a subsampling step by which the transformed 1-D data are classified into high and low frequency subbands, and the result is shown in Fig. 10. Then, the transformed image in Fig. 10 is transformed by the vertical lifting-based 5/3-wavelet transform. The algorithm of 1-D lifting-based vertical 5/3-wavelet DWT is just like the algorithm of the

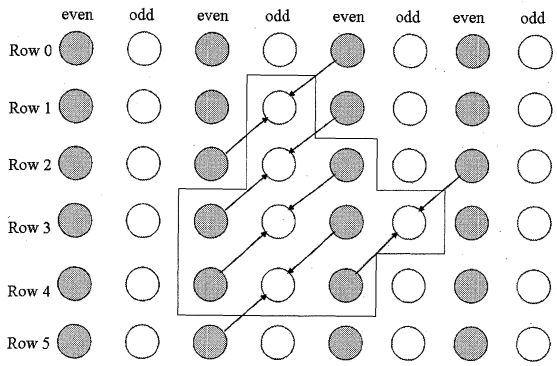


Fig. 11 The first prediction step of the 2-D shape-direction-adaptive DWT ($\theta = 45^o$) performed on an arbitrarily shaped segment.

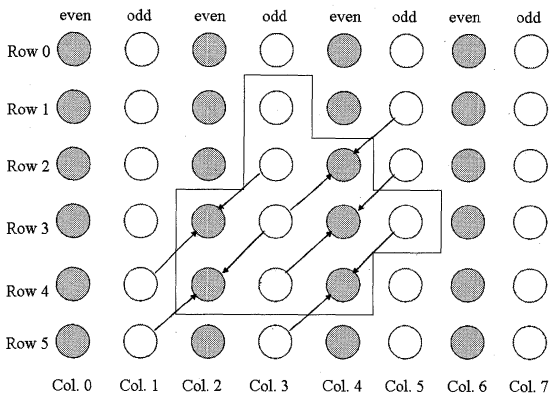


Fig. 12 The first update step of the 2-D shape-direction-adaptive DWT ($\theta = 45^o$) performed on an arbitrarily shaped segment.

1-D lifting-based horizontal 5/3-wavelet DWT.

2.4 The Proposed SDA-DWT

Shape-adaptive and direction-adaptive functionalities are two important improvements of DWT, and the experimental results in [1], [6], [9], and [11] show that they are very efficient for still image coding. ADL-DWT [1] and DA-DWT [6] were designed for processing rectangular images, so they can not process an arbitrarily shaped segment directly. On the other hand, SA-DWT [9] and the method proposed in [11] can process arbitrarily shaped object, but they do not offer the direction-adaptive functionality. In this paper, we proposed a new DWT which has both the shape-adaptive and direction-adaptive abilities, and we call it the shape-direction-adaptive DWT. The inputs of SDA-DWT, proposed in this paper, are the image containing the object to be transformed and the corresponding shape mask with or without partition, and the outputs are the transformed image of the object and the corresponding shape mask after SDA-DWT.

The proposed SDA-DWT can be described by using Figs. 11–16, and the same arbitrarily shaped segment in Figs. 8 and 9 and the 5/3-wavelet are used for illustration. Compared to SA-DWT step in Fig. 8, the corresponding SDA-DWT step is shown in Fig. 11. In Fig. 11, assume that

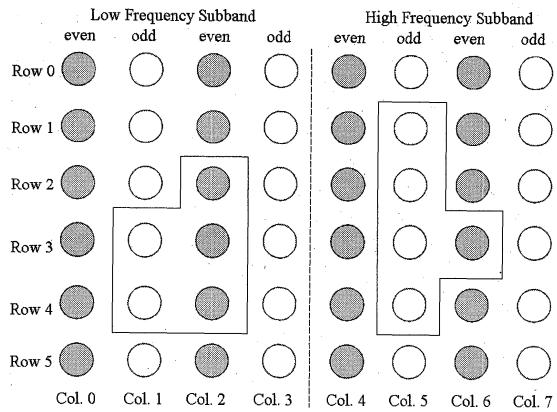


Fig. 13 The horizontal subsampling result of Fig. 12 in SDA-DWT.

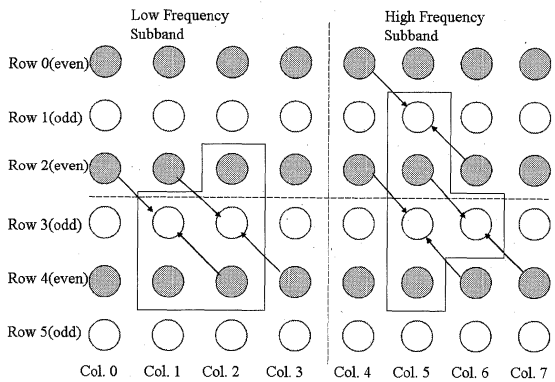


Fig. 14 The second prediction step of the 2-D SDA-DWT on an arbitrarily shaped segment.

the 45-degree direction is selected, so each odd pixel in the segment is predicted by two nearest even neighbors on the 45-degree line passing through this pixel. Then, each odd pixel is replaced by the residual obtained from subtracting the pixel value by the prediction value. If the prediction is good enough, the residual will be a small value. In the prediction step, the symmetric extension method is used for generating those even samples not in the segments, and the symmetric relation is about the line, passing through the odd pixel to be predicted, of 45 degree. According to Eq. (5), the update step in Fig. 12 is corresponding to the Fig. 9 of SA-DWT, and every even sample in the segment is updated by its two nearest odd neighbors (They already have been replaced by the residual values in the previous prediction step.) on the 45-degree line. After performing a pair of lifting steps (i.e. a prediction and an update steps), the transformed image is subsampled, and the result is shown in Fig. 13. The subsampling process is the same as the conventional horizontal subsampling method, and the subsampled coefficients are classified into the low-frequency subband and the high-frequency subband.

When the horizontal subsampling step is complete, the second part (corresponding to the vertical conventional 1-D DWT) of SDA-DWT begins from a prediction step (the second prediction step in SDA-DWT). Each odd sample in

columns of the segment is predicted by its left and right even neighbors on the 45-degree line compared to the vertical line (Fig. 14). Then, the second update step of SDA-DWT is performed on the even samples in columns of the segment (Fig. 15). Finally, a conventional subsampling along the vertical direction is performed on the coefficients in Fig. 15, and the image is transformed and divided into four subbands LL, LH, HL, and HH (Fig. 16). The symmetrical extension is used to generate the even samples and odd samples, not in the segment, for prediction and update, respectively. From Fig. 11 through Fig. 16, the one-level SDA-DWT is performed, and the LL subband can be used to be further transformed for multi-level SDA-DWT. The new shape mask is generated by subsampling the input shape mask along the horizontal and vertical directions, respectively.

Both ADL-DWT and DA-DWT partition an image into many small blocks. For texture features which are smaller than the smallest block size in the two methods, ADL-DWT and DA-DWT can not well exploit the correlation of the texture features in the small block. Moreover, even for large texture features, since the block locations are fixed, this makes the partition usually not optimal. Thus, some energy will be left in the high-frequency subband. The proposed method can handle any shaped segments at any location, so it can exploit the correlation of such textures and get better energy clustering to attain better compression efficiency.

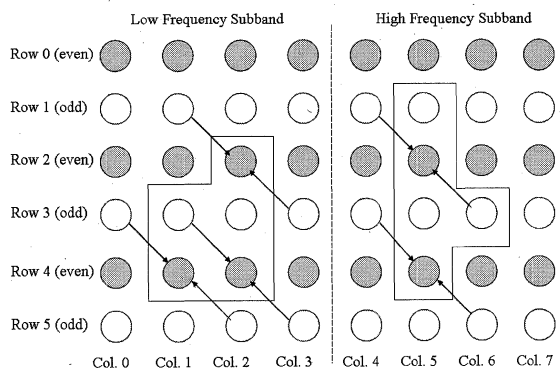


Fig. 15 The second update step of the 2-D SDA-DWT on an arbitrarily shaped segment.

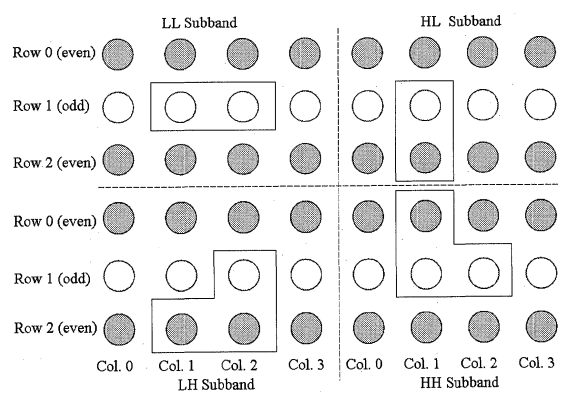


Fig. 16 The vertical subsampling result of Fig. 15 in SDA-DWT.

SA-DWT and the object-based SPECK are for object image compression, and they have the ability to process any shaped visual object. They do not have the texture-feature-size and fixed-block-location problems, but they use conventional (i.e. horizontal and vertical) directions for prediction and update. Lack of the capability of directional adaptability makes them unable to well exploit the spatial correlation of non-vertical and non-horizontal line textures. On the contrary, SDA-DWT can adapt the filter direction, according to the texture features in each of the partition segments of the interested object, for well exploiting the correlation and obtaining better compression efficiency.

In SDA-DWT, for handling the finite length data of 1-D wavelet transform, the symmetric extension of input data is used. The symmetric extension is effective and easy to implement, but, for the boundary between two partition segments, applying symmetric extension causes blocking effect for low bit-rate conditions. For such a problem, using the actual data at the extension points can alleviate the blocking effect. Periodic extension is another solution for finite length 1-D DWT computation, but it usually suffers from causing abrupt change at boundaries and needs more registers to implement.

3. Experimental Results

In this section, three test object images (Figs. 7, 18, 20) are used for simulation to evaluate the performance of SDA-DWT, SA-DWT and DA-DWT. The original sizes of test images 1 (Fig. 7) and 2 (Fig. 18) are 256-by-256 pixels, and the third test image (Fig. 20) is 128-by-128 pixels. Although the video frame size in MPEG-4 is 360-by-288, we choose square images in order to reduce the bits used for coding the paths in SPECK coding. For comparison, all methods (i.e. SA-DWT, DA-DWT, and SDA-DWT) use the same 5/3 wavelet, and both SA-DWT and SDA-DWT use symmetric extension for transform calculation while DA-DWT uses symmetric extension for transform calculation only on the boundary between the object image and background. For the partition boundaries in the object image, DA-DWT uses the practical values at the extension points. Here, we ignore the bits for side information (i.e. the partition of DA-DWT and the shape masks of SA-DWT and SDA-DWT) for simplification and focusing on the main problem. The decomposition-level decision in wavelet transform is important and difficult. For a suitable design of decomposition levels, energy clustering effect will make compression efficient. However, excessively many levels can not improve the overall compression efficiency, since the LL subband becomes a very small region that may degrade the overall compression efficiency. The suitable number of wavelet decomposition levels mainly depends on the image size, image content, and the coder/decoder used. In most cases, for a 512-by-512-pixel image, we select 3, 4, or 5 levels empirically. In this paper, 4 decomposition levels were used because the test images are small size. In the followings, PSNR (peak-signal-to-noise ratio) values and the lengths

Table 1 The bit numbers of the bit stream of each test object image after SPECK coding. (SDA^1 and SDA^2 represent SDA-DWT without object partition and with object partition, respectively. SA means SA-DWT and DA is DA-DWT)

Object image	Object 1	Object 2	Object 3
SDA^1	123,341 bits	244,729 bits	NA
SDA^2	NA	243,002 bits	67,726 bits
SA	158,530 bits	245,330 bits	71,556 bits
DA	NA	NA	77,209 bits

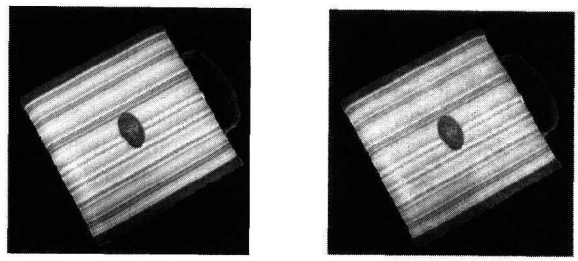
Table 2 The PSNR results for lossy compression of object image 1. (Object 1 contains 30,535 pixels)

Rate (bpp)	1.00	2.15	3.22
SDA	36.82 dB	42.75 dB	48.30 dB
SA	31.06 dB	36.87 dB	42.54 dB

of bit streams after SPECK coding are used as two performance measures. The PSNR calculation is based on a 256-by-256-pixel image (objects 1 and 2) or an 128-by-128-pixel image (object 3), and the bpp (bit/object pixel) calculation is based on the pixel number in an object image.

For the first object image (Fig. 7), our interested object is a suitcase covered with many line textures, and the object occupies 30,353 pixels in a 256-by-256-pixel image. SDA-DWT and SA-DWT are evaluated by compressing object-1 image. Since the orientations of lines in object 1 are almost the same, we do not partition the object into small segments, i.e. the whole visual object is a large segment. After performing 4-level SDA-DWT on the visual object, the transformed object image is coded by using the SPECK algorithm, and the resulted bit-stream can represent a compression file of the object image. The same procedures are performed on test image 1 except that SDA-DWT is replaced by SA-DWT, and we have another compression file of the object image by using SA-DWT. Table 1 shows the sizes (in bits) of each object image for each method, and it tells us that SDA-DWT is more efficient than SA-DWT is. The bit number of SDA-DWT compression file is about 77.8% size of the SA-DWT compression file. Table 2 shows that SDA-DWT outperforms SA-DWT up to 5.88 dB under 2.15-bpp (256×256 bits) condition. In this case, the performance of SDA-DWT is always better than that of SA-DWT because of the directional line textures on the object. For the characteristic of the textures on object 1, if we choose $+45^o$ direction in the prediction step of the 1-D 'horizontal' transform, the predicted values will very close to the actual values of odd pixels. Thus, much energy is clustered in the low-frequency subband, and that makes the wavelet transform very successful, which makes the overall compression scheme very efficiently. Two object-1 reconstruction images of SDA-DWT and SA-DWT, under 1-bpp condition (i.e. 30,353 bits), are shown in Fig. 17 for comparison. The reconstruction quality of SDA-DWT is obvious better than that of SA-DWT.

For the test image of object 2 (Fig. 18 (a)), SDA-DWT and SA-DWT are simulated and compared by their PSNR values and file sizes. The gray-level object-2 is segmented



(a) (b)

Fig. 17 The reconstruction images of object 1 under 1-bpp condition: (a) the result of SDA-DWT, (b) the result of SA-DWT. (Object 1 contains 30,353 pixels)

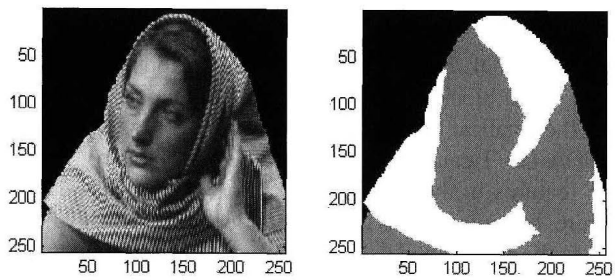


Fig. 18 Test object 2 and its shape mask with partition: (a) object-2 image in 256×256 frame, (b) object-2 shape mask with partition. (Object 2 contains 45,012 pixels)

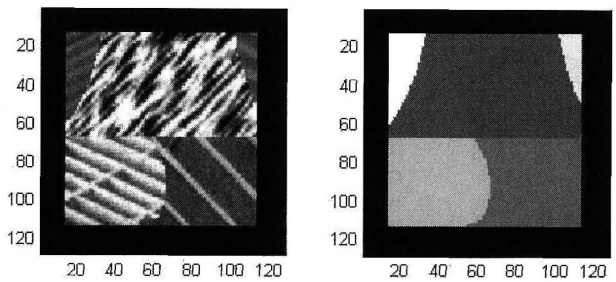
Table 3 The PSNR results for lossy compression of object image 2. (Object 2 contains 45,012 pixels. SDA^1 and SDA^2 represent SDA-DWT without object partition and with object partition, respectively)

Rate (bpp)	1.00	1.46	2.91
SDA^1	28.05 dB	28.88 dB	39.45 dB
SDA^2	28.11 dB	29.02 dB	39.45 dB
SA	27.52 dB	28.73 dB	39.30 dB

from the famous test image Barbara, and Fig. 18 (b) shows the shape mask of the visual object. Two cases are simulated for evaluating SDA-DWT. First, the whole object 2 without partition is used for simulation, and second, object 2 is partitioned into two parts (Fig. 18 (b), the white region and the gray part) for simulation. The partition shown in Fig. 18 (b) is an example for arbitrarily shaped partition which is not the optimal one. Table 1 shows that, for compression-file size, SDA-DWT with object-image partition is the most efficient case among these cases, SDA-DWT without object partition is second place, and SA-DWT is third place. SDA-DWT with object partition reduces 0.95% bit budget of SA-DWT's, and SDA-DWT without object partition reduces 0.24% bit-budget. On the other hand, the PSNR values in Table 3 show that SDA-DWT with partition has the best performance. The results show that for a texture rich (especially, non-horizontal or non-vertical edges) image, the performance of lossy compression can be enhanced by suitably partitioning the object image. The proposed method offers much flexibility for partition, since it can handle segments with any shape. The reconstruction object images of

(a) (b)

Fig. 19 The object-2 reconstruction images under 1.46-bpp condition: (a) the result of SDA-DWT with object partition according to Fig. 18 (b), (b) the result of SA-DWT.

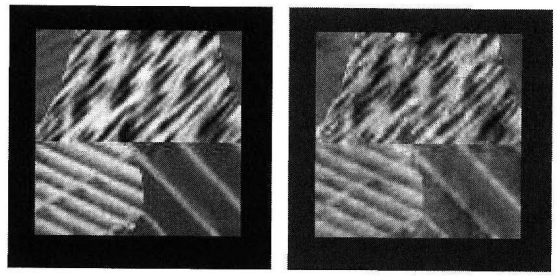


(a) (b)

Fig. 20 Test object 3 and its shape mask with partition: (a) object-3 image in 128 × 128 frame, (b) object-2 mask with partition. (Object 3 contains 10,000 pixels)

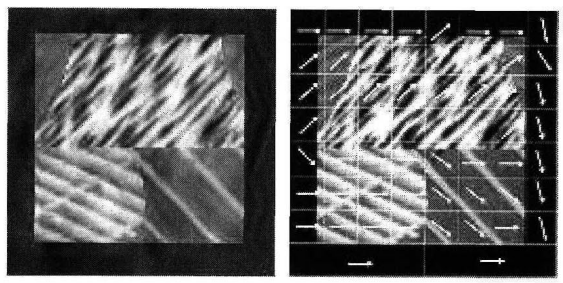
SDA-DWT with object partition and SA-DWT, under 1.46-bpp condition, are shown in Fig. 19. We also performed the experiments on the object images segmented from Lena, Claire, and Akiyo. Since these object images lack non-horizontal or non-vertical edges or the directions of textures are random, without suitable object partition, the performance of SDA-DWT and SA-DWT are almost the same.

For the third gray-level object image (Fig. 20 (a)), all the three methods (SA-DWT, DA-DWT, and SDA-DWT) are evaluated. The object-3 image (synthesized from the images in USC image database) contains 10,000 pixels in a 128-by-128-pixel area, and there are five different textures on the object. Hence, the object image is partitioned into 5 segments (Fig. 20 (b)) for SDA-DWT . Although object-3 image is rectangular, SDA-DWT can handle any shaped objects. DA-DWT is originally designed for processing a rectangular image, but object-3 image can be viewed as an squared 128-by-128-pixel image containing an object 3. DA-DWT partitions the object image into many small blocks (Fig. 22 (b)) to discover the texture direction which can not be seen in large scale. Table 1 shows that, for lossless compression, SDA-DWT uses the least amount of bits, and DA-DWT is the most bit consuming one. For the PSNR comparison, Tables 1 and 4 show that SDA-DWT outperforms SA-DWT up to 4.31 dB in PSNR under 1-bpp (bit / object pixel) condition, and reduces the bit-budget up to 5.66% (the base is 67,726 bits) for lossless compression. SDA-DWT also outperforms DA-DWT up to 5.44 dB in PSNR under 3.28-bpp condition, and reduces the bit-budget up to 14.0%. The reconstruction results under 1-bpp condition are shown in Figs. 21 and 22 (a). From the experiments of object 3, we understand that SA-DWT can not well exploit the correlation of the directional textures, so it has poor performance for this test object image. For DA-DWT, since its resolution is not high enough (the smallest partition block is 16-by-16), can not represent non-rectangular segment boundaries perfectly, and wastes coding bits on the object background; DA-DWT has the poorest performance for the special object-image.



(a) (b)

Fig. 21 The reconstruction object images, under 1-bpp condition: (a) the result of SDA-DWT, (b) the result of SA-DWT.



(a) (b)

Fig. 22 The reconstruction object image and the partition and direction in DA-DWT: (a) the reconstruction result under 1-bpp condition. (b) mask partition and block directions of partition used in DA-DWT.

Table 4 The PSNR results for lossy compression of object image 3. (Object 3 contains 10,000 pixels.)

Rate (bpp)	1.00	1.64	3.28	4.92
SDA[2]	22.41 dB	22.91 dB	33.29 dB	43.02 dB
SA	18.10 dB	22.58 dB	31.79 dB	40.55 dB
DA	17.53 dB	22.29 dB	27.85 dB	38.01 dB

4. Conclusions

In this paper we propose SDA-DWT, which can be used for arbitrarily shaped image segments, and whose direction of prediction and update are adaptive. From the experimental results, SDA-DWT has superior performance than SA-DWT or DA-DWT does for visual objects with non-horizontal or non-vertical edge textures. SDA-DWT can be applied to any wavelet-based application, although, in this paper, we only give the examples of the intra frame compression of

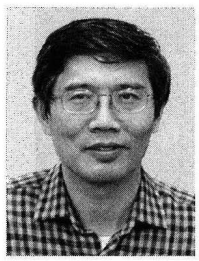
the object-based video compression in MPEG-4 standard. The extra costs of SDA-DWT compared to SA-DWT are the increased complexity and the storing and processing of the side information of the directions in each segment of the object image. For convenience, we focus on how to compress the partitioned still-object image while assuming that the partition of the object image has been done in this work. In order to achieve the optimal result, a good texture-segmentation method is necessary.

Acknowledgment

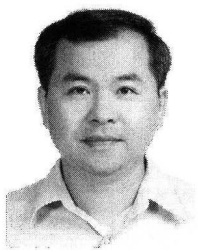
The authors would like to thank the anonymous reviewers for their comments that significantly helped improve this paper. This work was partially supported by the National Science Council of Taiwan, under Grant: NSC 96-2221-E-009-238.

References

[1] W. Ding, F. Wu, X. Wu, S. Li, and H. Li, "Adaptive directional lifting-based wavelet transform for image coding," IEEE Trans. Image Process., vol.16, no.2, pp.416–427, Feb. 2007.
[2] JPEG 2000 Image Coding System, ISO/IEC CD15444-1: 1999 (Version 1.0).
[3] I. Daubechies and W. Swendens, "Factoring wavelet transforms into lifting steps," J. Fourier Analysis and Applications, vol.4, no.3, pp.247–269, 1998.
[4] T. Acharya and C. Chakrabarti, "A survey on lifting-based discrete wavelet transform architectures," J. VLSI Signal Process., vol.42, pp.321–339, Feb. 2006.
[5] L. Yaroslavsky, "Fast signal sinc-interpolation and its applications in signal and image processing," IS&T/SPIE 14th Annu. Symp. Electronic Image 2003, Jan. 2002.
[6] C.-L. Chang and B. Girod, "Direction-adaptive discrete wavelet transform for image compression," IEEE Trans. Image Process., vol.16, no.5, pp.1289–1302, May 2007.
[7] S. Mallat, A Wavelet Tour of Signal Processing, 2nd ed., Academic Press, New York, USA, 1999.
[8] T. Sikora and B. Makai, "Shape-adaptive DCT for generic coding of video," IEEE Trans. Circuits Syst. Video Technol., vol.5, no.1, pp.59–62, Feb. 1995.
[9] S. Li and W. Li, "Shape-adaptive discrete wavelet transforms for arbitrarily shaped visual object coding," IEEE Trans. Circuits Syst., vol.10, no.5, pp.725–743, 2000.
[10] W.A. Pearlman, A. Islam, N. Nagaraj, and A. Said, "Efficient, low-complexity image coding with a set-partitioning embedded block coder," IEEE Trans. Circuits Syst. Video Technol., vol.14, no.11, pp.1219–1235, Nov. 2004.
[11] Z. Lu and W.A. Pearlman, "Wavelet coding of video object by object-based SPECK algorithm," Picture Coding Symposium (PCS 2001), pp.413–416, April 2001.



Sheng-Fuu Lin was born in Taiwan, the Republic of China, in 1954. He received the B.S. and M.S. degrees in mathematics from National Taiwan Normal University in 1976 and 1979, respectively, the second M.S. degree in computer science from the University of Maryland in 1985, and the Ph.D. degree in electrical engineering from the University of Illinois, Champaign, in 1988. Since 1988, he has been on the faculty of the Department of Electrical and Control Engineering at National Chiao Tung University, Hsinchu, Taiwan. His research interests include fuzzy theory, automatic target recognition, scheduling, image processing, and image recognition. Professor Lin is a member of the IEEE Control Society, Chinese Fuzzy System Association, and Chinese Automatic Control Society.



Chien-Kun Su was born in 1962. He received the B.S. degree from National Taiwan University in 1989 and M.S. degree from the University of Southern California in 1992. Currently, he is a Ph.D. student of the Electrical and Control Engineering Department at National Chiao Tung University in Hsinchu, Taiwan. He is also an instructor of the Electrical Engineering Department of Chung Hua University in Hsinchu, Taiwan. His research interests include image processing and computer vision.