Lifting Implementation of Reversible Deinterlacer

Takuma Ishida[1], Tatsuumi Soyama[1], Shogo Muramatsu[1], Hisakazu Kikuchi[1], and Tetsuro Kuge[2]
[1]Department of Electrical and Electronic Engineering, Niigata University Niigata 950-2181, Japan
[2]Science & Technical Research Laboratories, NHK 1-10-11 Kinuta, Setagaya-ku, Tokyo 157-8510, Japan

Abstract: In this work, an efficient lifting implementation of invertible deinterlacing is proposed. The invertible deinterlacing is a technique developed for intraframe-based video coding as a preprocessing. Unlike the conventional deinterlacing, it preserves the sampling density and has the invertibility. For a special selection of filters, it is shown that the deinterlacing can be implemented efficiently by an in-place computation. It is also shown that the deinterlacing can be combined with the lifting discrete wavelet transform (DWT) employed in JPEG2000. A bit modification of the original lifting DWT is shown to provide the simultaneous implementation of deinterlacing. This fact makes the proposed technique attractive for the application to Motion-JPEG2000. The inverse transform and the reversible lifting implementation are also discussed.

1. Introduction

Intraframe-based motion picture coding is known to superior to the interframe/field-based standards such as H.26x and MPEG-1/2/4 in terms of easiness of editing and system complexity. What demand such a technique are not only professional but also consumer applications. Motion-JPEG2000 (MJP2) is expected to be a new standard of intraframe-based technique as JPEG2000 part III [1,2]. For interlaced pictures such as NTSC signals [3,4], the intraframe-based approach requires field interleaving so that the still picture coding is directly applicable. This process, however, causes horizontal comb-tooth artifacts at edges of the moving objects [3–6]. As a previous work, in order to solve the problem we developed a new class of deinterlacing and generalized the field-interleaving [6]. Deinterlacing is a technique to convert the scanning format of motion pictures from interlaced to progressive scanning [3–5]. Historically, deinterlacing has been used to avoid the aliasing and flicker artifacts when displaying the broadcasted TV, or to convert video formats between different standards, for instance, from NTSC signal to VGA signal. Unlike the conventional deinterlacing, the proposed deinterlacing preserves the sampling density and has the invertibility so as to be applied as preprocessing of coding. The design method of the invertible deinterlacing is given in the article [6].

In this work, we give an efficient implementation of the deinterlacing for a special case. The proposed technique has a main feature of lifting implementation. MJP2 is most likely to be the application. Thus, it is worth investigating to evaluate the overhead required to apply the deinterlacer to JPEG2000. JPEG2000 is known to employ the lifting discrete wavelet transform (DWT), which requires only simple operation [2]. However, the deinterlacing followed by DWT is redundant manipulation. In order to reduce the overhead, we propose to combine the deinterlacing with the lifting implementation of DWT in JPEG2000. In consequence, it is shown that both deinterlacing and DWT can be computed simultaneously, and only a bit modification for DWT is required. The inverse transform and the reversible realization are also discussed.

In this work, the following notations are used. $\mathbf{z}$: a 3×1 vector which consists of variables in a 3-D z-domain, that is, $\mathbf{z} = (z_0, z_1, z_2)^T$. For progressive arrays, z_0, z_1 and z_2 denote the variables for the temporal, vertical and horizontal directions, respectively. In this case, we express $\mathbf{z}$ as $(z_\mathrm{T}, z_\mathrm{V}, z_\mathrm{H})^T$. $\mathbf{z}^\mathbf{n}$: the product defined by $\mathbf{z}^\mathbf{n} = z_0^{n_0} z_1^{n_1} z_2^{n_2}$, where $\mathbf{n}$ is a 3×1 integer vector, and n_k denotes the k-th element of $\mathbf{n}$. $\mathcal{L}(\mathbf{V})$: the set of sample points given by $\mathbf{Vn}$ for all 3×1 integer vectors $\mathbf{n}$, where $\mathbf{V}$ is a 3×3 non-singular matrix [7,8]. $J(\mathbf{V})$: the absolute determinant of $\mathbf{V}$. $\rho(\mathbf{V})$: the inverse of the absolute determinant of a sampling matrix $\mathbf{V}$, that is, the sampling density.

2. Review of invertible deinterlacing

As a previous work, we proposed a new class of deinterlacing for intraframe-based video codec system such as MJP2 [6]. This deinterlacing preserves sampling density and has invertibility. In this section, let us review this technique as a preliminary. It should be noted that the original array $X(\mathbf{z})$ is assumed to be sampled on the lattice $\mathcal{L}(\mathbf{V})$ generated by the sampling matrix $\mathbf{V} = \begin{pmatrix} P_\mathrm{T} & P_\mathrm{T} & 0 \\ -P_\mathrm{V} & P_\mathrm{V} & 0 \\ 0 & 0 & P_\mathrm{H} \end{pmatrix}$, where P_T, P_V and P_H are the temporal period between successive fields, the vertical and horizontal sampling periods in a frame, respectively. Figure 1 shows the sampling lattice of the original array $X(\mathbf{z})$. The sampling density $\rho(\mathbf{V})$ is given as $1/2P_\mathrm{T}P_\mathrm{V}P_\mathrm{H}$.

2.1 Deinterlacing with sampling density preservation

Figure 2 shows the basic structure of a deinterlacer with sampling density preservation, where the circles including $\uparrow\mathbf{Q}$ and $\downarrow\mathbf{R}$ denote the upsampler with a factor $\mathbf{Q}$ and downsampler with a factor $\mathbf{R}$, respectively [3,4,7,9]. The upsampler converts the interlaced video array $X(\mathbf{z})$ into the non-interlaced one. For the sampling matrix $\mathbf{V}$, the factor $\mathbf{Q}$ has to be $\begin{pmatrix} 1 & 1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$. The downsampling is applied so that the overall output array $Y(\mathbf{z})$ has the same density as the original, where the ratio $J(\mathbf{R})$ has to be 2 and the lattice orthogonal-

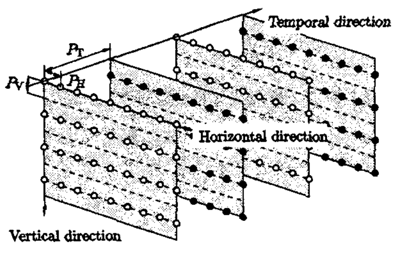


Figure 1. Interlaced scanning with the sampling lattice $\mathcal{L}(\mathbf{V})$. The white and black circles are the sample points on even and odd lines, respectively.

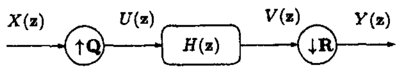


Figure 2. Basic structure of deinterlacer

ity has to be kept. $H(\mathbf{z})$ is a 3-D filter, which removes the imaging caused by the upsampler and avoids the aliasing to be caused by the downsampler. One choice of $\mathbf{R}$ is given by $\mathbf{R} = \left(\begin{smallmatrix}2&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right)$. In this case, $V(\mathbf{z})$ is temporally downsampled. Then, the sampling matrix of $Y(\mathbf{z})$ becomes $\mathbf{V}' = \mathbf{V}\mathbf{Q}^{-1}\mathbf{R} = \left(\begin{smallmatrix}2P_T&0&0\\0&P_V&0\\0&0&P_H\end{smallmatrix}\right)$. The diagonal form implies the orthogonality of the sampling lattice $\mathcal{L}(\mathbf{V}')$. The sampling density $\rho(\mathbf{V}')$ results in $1/2P_TP_VP_H$, which is identical to the original. The corresponding sampling lattice is shown in Fig. 3.

2.2 Reinterlacing with sampling density preservation

For video codec systems, the deinterlaced array $Y(\mathbf{z})$ is encoded, transmitted and then decoded. Especially for the high bit-rate decoding, the interlaced video source is expected to be reconstructed. To achieve this, we have introduced the inverse converter as *reinterlacer* [6]. Figure 4 illustrates the basic structure of the

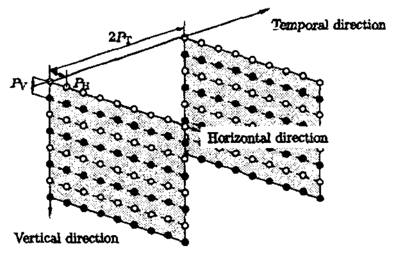


Figure 3. Sampling lattice $\mathcal{L}(\mathbf{V}')$ of deinterlaced array with temporal decimation.

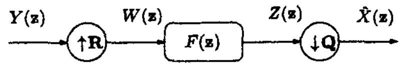


Figure 4. Basic structure of reinterlacer

reinterlacer. This enables us to obtain an array $X(\mathbf{z})$ on the same sampling lattice as the original $\mathbf{V}$, because $\mathbf{V}'\mathbf{R}^{-1}\mathbf{Q} = \mathbf{V}$. It can be proved that if the following conditions are satisfied, the source $X(\mathbf{z})$ is perfectly reconstructed, except for scaling and delay [6].

Alias Free Condition:

$$F(\mathbf{z}) = \mathbf{z}^{-\left(\begin{smallmatrix}2S_T+1\\2S_V\\S_H\end{smallmatrix}\right)} KH\left(W_2^{(0\ 1\ 0)}\mathbf{z}\right), \tag{1}$$

where K, S_T, S_V and S_H are integers. This relation implies that $F(\mathbf{z})$ is the vertically π-modulated version of $H(\mathbf{z})$ with some odd number of temporal delay $2S_T+1$, even number of vertical shift $2S_V$ and an arbitrary horizontal shift S_H.

Distortion Free Condition:

$$K\left\{E_{\mathbf{m}_0}(\mathbf{z})E_{\mathbf{m}_1}(\mathbf{z}) - \mathbf{z}^{-\left(\begin{smallmatrix}0\\1\\0\end{smallmatrix}\right)}E_{\mathbf{m}_2}(\mathbf{z})E_{\mathbf{m}_3}(\mathbf{z})\right\} = \mathbf{z}^{-\mathbf{d}} \tag{2}$$

for any 3×1 integer vector $\mathbf{d}$, where $E_{\mathbf{m}_k}(\mathbf{z})$ be the k-th type-I polyphase component of $H(\mathbf{z})$ with the factor $\mathbf{S} = \mathrm{diag}(2,2,1)$ such that

$$H(\mathbf{z}) = \sum_{k=0}^{3} \mathbf{z}^{-\mathbf{m}_k} E_{\mathbf{m}_k}(\mathbf{z}^{\mathbf{S}}),$$
$$\mathbf{m}_0 = \left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right), \mathbf{m}_1 = \left(\begin{smallmatrix}1\\0\\0\end{smallmatrix}\right), \mathbf{m}_2 = \left(\begin{smallmatrix}0\\1\\0\end{smallmatrix}\right), \mathbf{m}_3 = \left(\begin{smallmatrix}1\\1\\0\end{smallmatrix}\right). \tag{3}$$

3. In-place computation

Let us show a special pair of invertible deinterlacing and reinterlacing filters:

$$H(\mathbf{z}) = 1 + \frac{1}{2}z_T^{-1} + \frac{1}{4}(z_V^{1} + z_V^{-1}) \tag{4}$$

$$F(\mathbf{z}) = 2 + z_T^{-1} - \frac{1}{2}(z_V^{1} + z_V^{-1}) \tag{5}$$

These filters are designed for satisfying the perfect reconstruction conditions [6]. In addition, the following constraints are applied in order to give some preferable properties: normalized amplitude to keep the illuminace, regularity to avoid the checker board effect [10,11], and vertical symmetry to apply the symmetric border extension method [10, 12]. Note that all of the coefficients are power of two.

For the pair of these filters, deinterlacing and reinterlacing can be efficiently implemented as illustrated in Fig. 5, where the white, black and gray circles denote even line, odd line of the interlaced array and odd line of the deinterlaced array, respectively. It can be noticed that an in-place computation is available. These structures avoid the redundant computations caused by downsampling and upsampling. Figure 6 shows the corresponding lifting structures. These procedures are composed of one predict with simple Haar type 2-tap filters. These lifting transforms are applied to the vertical direction.

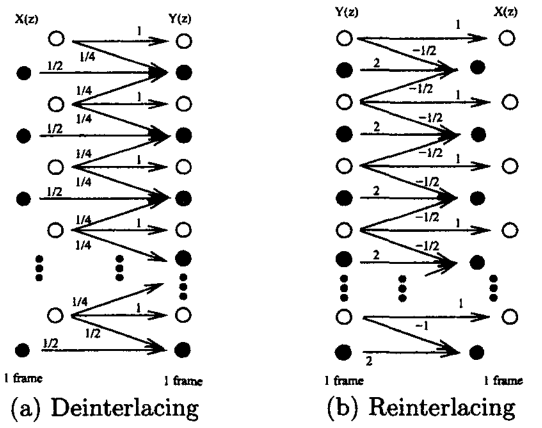


Figure 5. In-place computation of deinterlacing and its inverse, where only vertical-temporal plane is shown.

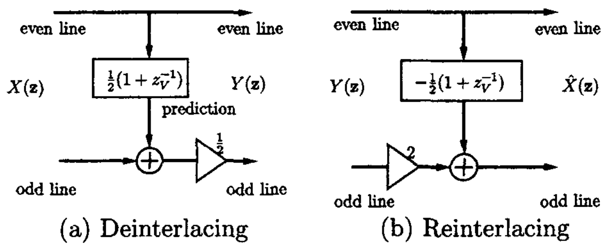


Figure 6. Efficient lifting implementation.

4. Lifting implementation with DWT

In the previous section, we consider implementing only deinterlacing and reinterlacing by using lifting structure. In this section, we show that the deinterlacing and reinterlacing can be combined with DWT in JPEG2000 and implemented efficiently.

4.1 DWT for JPEG2000

Let us explain the lifting implementation of DWT employed in JPEG2000 in brief for the following discussions. Two types of transforms are standardized in JPEG2000 [2]. One is the 5/3 transform, which is shown in Fig. 7, and the other is the 9/7 transform. One of the main features of JPEG2000 is the support of lossless compression mode. The lossless compression is realized by the reversible 5/3 transform where the rounding operations are applied to output of each prediction and

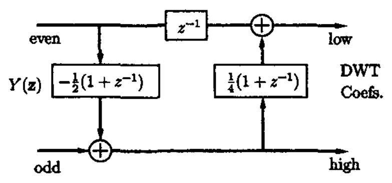


Figure 7. Lifting implementation of 5/3 DWT.

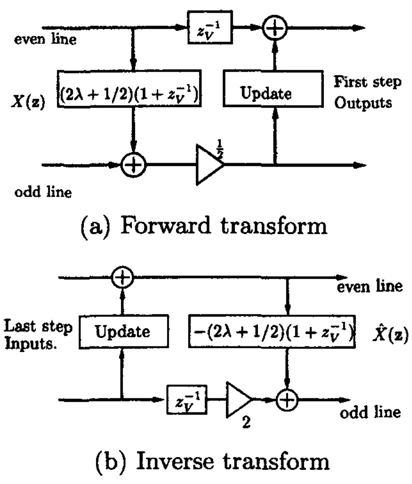


Figure 8. Deinterlacing with the first vertical stage of lifting DWT and its inverse.

update filter in Fig. 7.

4.2 Proposed structure

Note that all of the prediction and update filters in 5/3 and 9/7 transforms are Haar type 2-tap filters with some different scaling factors. The first prediction of the first vertical lifting stage in DWT can be combined with the deinterlacing shown in Fig. 6 (a). It is possible to calculate both DWT and deinterlacing simultaneously. Similar discussion holds for the inverse transform and reinterlacing. Figure 8 shows the proposed methods. The scaling factor λ of the prediction is expressed as $\lambda = -1/2$ for the 5/3 transform. For the forward transform, the coefficients of the first prediction are modified and samples on odd lines are shifted their values to right by one bit. Similarly, for the inverse transform the coefficients of the last prediction of lifting are replaced and samples on odd lines are shifted their values to left. Therefore, some simple modification on the lifting DWT enables us to implement the invertible deinterlacing and reinterlacing. For the original lifting DWT, all we have to do is to add one shift operation and modify one prediction.

Architecture: Figure 9 shows an example of deinterlacer-embedded DWT architecture, which is based on that proposed in [13]. The architecture provides simultaneous implementation of deinterlacing and lifting-based DWT by incorporating the coefficients of deinterlacing into the DWT's prediction filter. In Fig.9, the 5/3 transform uses the coefficients $-\frac{1}{2}$ and $\frac{1}{4}$, and the 9/7 transform uses α, β, γ and δ defined in the standard. The proposed architecture can also carry out only the lifting-based DWT by choosing the filter coefficients for the usual transform.

Performance evaluation: This evaluation will be carried out by using the synthesis results obtained from VHDL models of lifting architectures. Here the proposed archi-

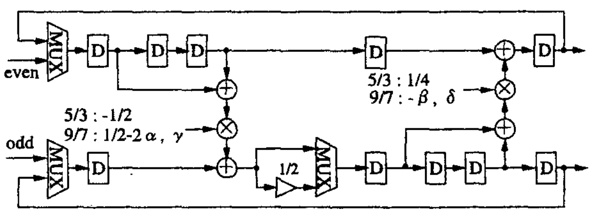
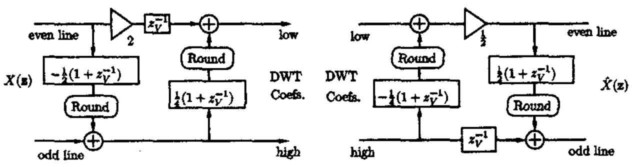


Figure 9. Architecture of DWT lifting with deinterlacing.



(a) Forward transform (b) Inverse transform

Figure 10. Reversible lifting implementation of deinterlacing with 5/3 transform and its inverse.

tecture is compared to that for the conventional lifting-based DWT. The optimization processes are applied for minimizing the area. In this evaluation, Xilinx XCV300 is selected as a target device. The proposed architecture is modeled such that the input bits are set to 32 bits, where the integer part is 16 bits and the decimal part is 16 bits. Synopsys Design Analyzer version 2000.05, which is provided by VDEC [14], is used for the synthesis, and Xilinx Design Manager version 3.3.08i is used for PAR process.

Controllers, which are not shown in Fig.9, are also taken into account. In the case of the deinterlacer-embedded architecture, the number of equivalent gates increases only 0.8%. The proposed architecture requires only 217 gate-increase, whereas the number of equivalent gates results in 1,670 when deinterlacing is implemented as a separate module.

4.3 Reversible implementation

Lossless lifting implementation is realized by rounding the outputs of each lifting filter to integer values. This fact implies that the reversible implementation of deinterlacing is possible. Figure 10 illustrates the reversible lifting implementation. From Fig. 8(a), it is noticed that the samples on odd lines are halved, which disturbs the reversibility. In order to achieve the reversible transform, the coefficients of deinterlacing filter $H(\mathbf{z})$ are doubled and those of reinterlacing filter $F(\mathbf{z})$ are halved in the structure in Fig. 10. The modified filters $H(\mathbf{z})$ and $F(\mathbf{z})$ are as follows: $H(\mathbf{z}) = 2 + z_T^{-1} + \frac{1}{2}(z_V^1 + z_V^{-1})$, $F(\mathbf{z}) = 1 + \frac{1}{2}z_T^{-1} - \frac{1}{4}(z_V^1 + z_V^{-1})$. Samples on odd lines are non-weighted. Note that the filter coefficients in the proposed reversible lifting method are identical to those of the original transform shown in Fig. 7. Difference is recognized only on the scaling for samples on even lines. This is a quite attractive point of the proposed method.

5. conclusion

In this work, a lifting implementation of invertible deinterlacing was proposed. For a special pair of filters, it was shown that the deinterlacing can be implemented efficiently by means of the in-place computation. In addition, it was also shown that the deinterlacing can be combined with the first vertical stage of lifting DWT in JPEG2000. By adding one bit-shift operation and modifying one prediction filter, the lifting DWT is shown to provide simultaneous implementation of deinterlacing. This fact makes the proposed technique attractive for the application to Motion-JPEG2000. The inverse transform and the reversible realization were also discussed. The filter coefficients in the proposed reversible lifting is shown to be identical to those of the original DWT lifting, where only a bit-shift operation is appended.

Acknowledgement

This work was in part supported by the Grand-in-Aid for Scientific Research No.14750283 from Society for the Promotion of Science and Culture of Japan.

References

[1] T. Fukuoka, K. Katoh, S. Kimura, K. Hosaka, and A. Leung, "Motion-JPEG2000 standardization and target market," in *Proc. IEEE ICIP*, no. TA0208, Sept. 2000.

[2] D.S. Taubman and M. W. Marcellin, *JPEG2000 Image Compression Fundamentals, Standards and Practice*, Kluwer Academic Publishers, 2002.

[3] Y. Wang, J. Ostermann, and Y. Zhang, *Video Processing and Communications*, Prentice-Hall, 2002.

[4] A. Murat Tekalp, *Digital Video Processing*, Prentice Hall, Inc., 1995.

[5] G. de Haan and E. B. Bellers, "Deinterlacing-an overview," in *Proc. IEEE*, Sept. 1998, vol. 86, pp. 1837–1857.

[6] S. Muramatsu, T.Ishida, and H. Kikuchi, "A design method of invertible de-interlacer with sampling density preservation," in *IEEE Proc. of ICASSP*, no. IMDSP-L03,01, 2002.

[7] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*, Prentice Hall, Englewood Cliffs, 1993.

[8] S. Muramatsu, A. Yamada, and H. Kiya, "A design method of multidimensional linear-phase paraunitary filter banks with a lattice structure," *IEEE Trans. Signal Processing*, vol. 47, pp. 690–700, Mar. 1999.

[9] S. Muramatsu and H. Kiya, "Multidimensional parallel processing methods for rational sampling lattice alteration," *Proc. IEEE ISCAS*, vol. 1, pp. 756–759, Apr. 1995.

[10] G. Strang and T. Nguyen, *Wavelets and Filter Banks*, Wellesley-Cambridge Press, 1996.

[11] S. Muramatsu Y. Harada and H. Kiya, "Multidimensional multirate filter and filter bank without checkerboard effect," *IEICE Trans. Fundamentals*, vol. E81-A, no. 8, pp. 1607–1615, Aug. 1998.

[12] K. Nishikawa H. Kiya and M. Iwahashi, "A development of symmetric extension method for subband image coding," *IEEE Trans. Image Processing*, vol. 3, no. 1, pp. 78–81, Jan. 1994.

[13] C.J. Lian, K.F. Chen, H.H. Chen and L.G. Chen, "Lifting based discrete wavelet transform architecture for JPEG2000," *Proc. IEEE ISCAS*, pp. 445–448, 2001.

[14] VDEC : "http://www.vdec.u-tokyo.ac.jp/".