PAPER

Platform-Based Design for the Low Complexity and High Performance De-Interlacing System

Tsung-Han TSAI^{†a)}, Member and Hsueh-Liang LIN[†], Nonmember

SUMMARY With the development of digital TV system, how to display the NTSC signal in digital TV system is a problem. De-interlacing is an algorithm to solve it. In previous papers, using motion compensation (MC) method for de-interlacing needs lots of computation complexity and it is not easy to implement in hardware. In this paper, a content adaptive de-interlacing algorithm is proposed. Our algorithm is based on the motion adaptive (MA) method which combines the advantages of intra-field and inter-field method. We propose a block type decision mechanism to predict the video content instead of a blind processing with MC method throughout the entire frame. Additionally, in intra-field method, we propose the edge-base adaptive weight average (EAWA) method to achieve a better performance and smooth the edge and stripe. In order to demonstrate our algorithm, we implement the de-interlacing system on the DSP platform with thorough complexity analysis. Compared to MC method, we not only achieve higher video quality in objective and subjective view, but also consume lower computation power. From the profiling on CPU runtime analysis, the proposed algorithm is only one-fifth of MC method. At the DSP demonstration board, the saving ratio is about 54% to 96%.

key words: *de-interlace, de-interlacing, intra-field interpolation method, inter-field interpolation method, motion adaptive method, motion compensation method*

1. Introduction

De-interlacing converts interlaced video sequences into progressive ones for displaying on progressive devices such as LCD, progressive CRT, and projection TV. It is a picture format conversion from interlaced to progressive picture, and has been widely used to reduce the visual effects on those artefacts. It is developed to recover all the defects appearing in the interlaced to progressive conversion.

Various de-interlacing techniques have been proposed in last few decades. Those can be categorized into two categories: intra-field method and inter-field method. The intra-field method [1]–[5] is used with spatial filtering method and edge-directional interpolation method such as “Edge Line Average” (ELA) [2], one of well-known spatial domain interpolation technique. Chang [3] proposed an extended intelligent edge-based line average method which finds the dominant edge to avoid the wrong interpolated value from background area. In [4], a directional interpolation instead of the traditional ELA is used to enhance the performance. The similar idea is used in [5] for the field-sequential stereoscopic video usage. In [6], an edge-dependent interpolation method with dedicated consideration on edge patterns

is proposed. The advantages of intra-field methods are low computational complexity and hardware cost.

The inter-field methods include motion adaptive filtering and motion compensation. The first inter-field method is developed to merge the two fields into a frame. Until 1999, Sugiyama [7] used motion compensation (MC) for de-interlacing. MC is applied to the temporal correlation of video sequence and find the most similar pixel from the neighbourhood fields to be interpolated. Kwon [8] also used MC method for de-interlacing and even used half pixel accuracy to refine the motion vector. Chang [9] proposed a same parity four field local MC de-interlacing method to utilize the characteristic of interlaced video on motion estimation. Although MC method can achieve better de-interlacing quality, it needs a motion estimation engine and consumes a lot of computation power. Consequently, the computational complexity and hardware cost are the main concerns in MC de-interlacing algorithm.

Currently, motion adaptive method (MA) [10]–[14] is treated as one of inter-field methods and the main stream of the de-interlacing method. Motion adaptive method relies on accurate motion detection [11]. Some inter-field methods even use motion compensation to find the similar block from neighbour field, consequently it needs more computation power and requires a motion estimation engine. In MA method, motion detection plays an important role to combine intra-field method and inter-field method; but how to perform accurate motion detection is the key point.

De-interlacing technique is certainly quality-oriented, but it is also necessary to consider the complexity and implementation issue. To enhance the video quality, many algorithms are intuitively constructed based on MC method. Although the block moving information is actually reliable for recovering the video, exhaustive motion estimation/compensation consumes enormous computation power. Recently, the de-interlacing designs are realized as a single chip with ASIC approach [15], [16]. Some of them even accomplish the de-interlacing engine as a high complexity design. In recent trends, the displaying IC should be an integrated system and de-interlacing engine is only a component of it. How to reduce the de-interlacing complexity and cost but still maintain the video quality is very crucial. In point of a displaying system, a platform-based design for de-interlacing issue should be more valuable, especially for the requirements of low cost and easy implementation on consumer application.

In this paper, we propose a novel content adaptive

Manuscript received April 10, 2008.

Manuscript revised July 25, 2008.

[†]The authors are with the Department of Electrical Engineering, National Central University, Chung-Li, Taiwan, R.O.C.

a) E-mail: han@ee.ncu.edu.tw

DOI: 10.1093/ietisy/e91-d.12.2784

de-interlacing technique. We use a block type decision to predict the video content instead of a blind processing by MC method for overall frame. Our block type decision not only avoids non-adaptive area to be interpolated by MC method, but also selects the most adaptive interpolation method for use. Besides, we also propose a new intra-field method which has a better performance on smoothing the edge and stripe. Experimental result shows that we can save large computational power and still get high video quality than other de-interlacing algorithms. In order to prove our algorithm with real-time constrain, we use an ATEME IEKC64x platform to realize it. We can capture the field-based video and de-interlace it to progressive type then display it on LCD monitor. This paper is organized as follows. The previous methods and some design issues are listed in Sect. 2. Our proposed de-interlacing method is shown in Sect. 3. Experimental results are shown in Sect. 4. Section 5 will introduce our hardware platform first and describe the implementation issue. We will give some conclusions in Sect. 6.

2. De-Interlacing Design Issues

De-interlacing methods can be mainly characterized into two categories: intra-field method and inter-field method. Because a straightforward concept occurred in edge-base line average (ELA), we first investigate this design and then discuss the design issue for inter-field methods.

2.1 Design Issues on Intra-Field Method

Generally, intra-field method is the most popular method for de-interlacing algorithm. It is widely used in software implementation because it requires less computational power and only needs several line buffers. Line doubling is first proposed for intra-field method. This method repeats every line again in current field. However, line doubling method is not adaptive to be used in texture area. In recent years, ELA was proposed to solve this problem. In order to reduce the flicker effect from line doubling, ELA uses three pixels in the previous scan line and the next scan line to detect the edge in the field as shown in Fig. 1.

For ELA, first is to find the difference D . D is calculated to get the correlation of the pixel in Eq. (1):

$$D(k) = \text{abs}(x(i-1, j+k) - x(i+1, j-k)) \quad (1)$$

k is a value which equals to +1, 0 or -1. Second is to find the minimum difference D as the highest correlation in Eq. (2):

$$k = \arg \min_{-1 \leq k \leq 1} D(k) \quad (2)$$

According to k value, we can find the interpolated value in Eq. (3).

$$x(i, j) = (x(i-1, j+k) + x(i+1, j-k))/2 \quad (3)$$

Although ELA can reduce some defects such as line

Fig. 1 Edge-base line average (ELA).

Fig. 2 Motion compensation (MC) method.

blurred, this method only has the same resolution as current processing field. In addition, the way to the success of ELA is an accurate estimation on edge direction. However, some interpolated values are unpleasant results due to the noise and weak edge. Because ELA only depends on the up and down scan lines, it can not detect the edge existence well.

2.2 Design Issues on Inter-Field Method

Inter-field method uses the temporal correlation between the fields and finds the more adaptive value to be interpolated. Field merge method is the first proposed algorithm for inter-field method. However, this method needs one field buffer, instead of only several line buffers needed in intra-field method. Compared with intra-field method, inter-field method can get better video quality in the stationary area. In recent years, some algorithms use MC method for inter-field method to get better video quality in moving area. It utilizes the temporal correlation of video sequence and finds the most similar pixel from the neighbourhood fields to be interpolated, as illustrated in Fig. 2. However, MC method needs a motion estimation engine which consumes a lot of computational power.

The motion adaptive method can be classified into inter-field method but with more consideration. It basically consists of intra-field method, inter-field method and motion detection blocks. Since intra-field method can get better video quality in moving area and inter-field method is more adaptive for stationary area, MA method combines the advantages of both intra-field and inter-field method. Motion detection provides motion information which helps users to distinguish the area moving or stationary area. It plays an

important role on the performance of MA method.

MA method seems to have covered the drawbacks on de-interlacing method, but two intuitive issues need further considerations. First, in order to get better video quality, MC method is widely used as the inter-field method in MA method. If motion detection decides the interpolation method with intra-field method, the heavy computational power on motion estimation in inter-field method is wasted. How to make a precise judgement on intra- or inter-based interpolation is challenging. Second, some scenario on video will confuse the MA method. For example, scene change is always occurred in video. Especially, some high-motion video contains frequent scene changes on the whole frame or even on some dominated objects.

3. Proposed De-Interlacing Algorithm

3.1 Overall Design

In order to get better video quality, MC method is widely used to replace inter-field method in MA method. Obviously, MC method needs more computational power than other inter-field methods. Some experimental results show that only a portion of pixels are adaptive to be interpolated by MC method. For example, current processing field with scene change or new objects moved in will cause the feathering effect on video. In these kinds of area, MC method can not find the similar pixel value from the neighbour fields. With this knowledge we propose a content adaptive predictor to avoid the wasted computation on motion estimation in these areas.

The block diagram of the proposed algorithm is shown in Fig. 3. It consists of intra-field method, inter-field method and block type decision. For inter-field method, we use two-field MC method. It utilizes motion estimation to find the most similar block in the previous same field and calculate the motion vector. Then it interpolates the value from the previous same field by tracing motion vector. We also perform previous field insert (PFI) to find the best interpolated value for stationary or background area. If the difference value after motion detection is smaller than the threshold, we will treat the area as the stationary area. Then we can find the most adaptive value to be interpolated from the previous opposite field.

In our algorithm, two critical mechanisms are developed. For intra-field method, we propose the edge-base adaptive weight average (EAWA) method. Compared with traditionally ELA, we can detect the edge more accurately, and it is more adaptive for texture area. Block type decision is the other important idea in our proposed algorithm. It can circumvent the limitations on MC method and predict the video content in the early stage. With this mechanism, we can precisely classify the inter- or intra-field method without the blind motion estimation on every pixel. Details for these two designs are described below.

Fig. 3 Block diagram of the proposed algorithm.

Fig. 4 The horizontal edge detection using Prewitt horizontal edge mask.

Fig. 5 The vertical edge detection, where the pixels *a* to *h* are generated with half-pixel interpolation.

3.2 Edge-Base Adaptive Weight Average (EAWA)

1): Horizontal & vertical edge detection:

As shown in Fig. 4 and Fig. 5, we use two 3×3 Prewitt operators for horizontal and vertical edge detection [17]. After using these two operators, the pixel can be divided into three areas: flat, dominated edge, and ELA. First, we use horizontal edge detection to ensure the pixel is in flat area, as in Eq. (4).

$$H(i, j) = \text{abs}(x(i-1, j-1) + x(i-1, j) + x(i-1, j+1) - x(i+1, j-1) - x(i+1, j) - x(i+1, j+1)) \quad (4)$$

We further need the horizontal threshold th_H to decide it. If $H(i, j)$ is smaller than a threshold th_H , this pixel is located in flat area. In our experiment, the value of th_H is 10.

If the pixel is not in flat area, we perform vertical edge detection. For vertical edge detection, first we import half-pixel to increase the accuracy. Such as the pixels *a* to *h* in Fig. 5, we use two close pixels to interpolate the half-pixel.

For example, the sample a in Fig. 5 can be derived as the average of sample $x(i-1, j-2)$ and $x(i-1, j-1)$. Then we detect the five edge directions. We use Prewitt vertical edge mask to calculate five vertical coefficients as in Eq. (5).

$$\begin{aligned} V(j-1) &= \text{abs}(a + g - b - h) \\ V(b) &= \text{abs}(x(i-1, j-1) + x(i+1, j) \\ &\quad - x(i-1, j) - x(i+1, j+1)) \\ V(j) &= \text{abs}(b + f - c - g) \\ V(c) &= \text{abs}(x(i-1, j) + x(i+1, j-1) \\ &\quad - x(i-1, j+1) - x(i+1, j)) \\ V(j+1) &= \text{abs}(c + e - d - f) \end{aligned} \quad (5)$$

Additionally, the vertical threshold th_V is used to perform the vertical edge detection. This value is just the average of the five vertical coefficients plus an offset value of 30. We compare five vertical coefficients to vertical threshold value. If one of five vertical coefficients is bigger than th_V , it means there is one or several dominated edges in five directions. In this condition, we use our “Edge Weight Function” described later to interpolate value. Otherwise it means that all vertical coefficients are smaller than vertical threshold value and consequently there is no flat area or no dominated edge. In that case we use “ELA” to interpolate value.

2): *Edge weight function:*

As illustrated in Fig. 5, first we calculate the difference value (*diff*) of the five directions as in Eq. (6).

$$\begin{aligned} \text{diff}_1 &= \text{abs}(x(i-1, j-1) - x(i+1, j+1)) \\ \text{diff}_2 &= \text{abs}(b - g) \\ \text{diff}_3 &= \text{abs}(x(i-1, j) - x(i+1, j)) \\ \text{diff}_4 &= \text{abs}(c - f) \\ \text{diff}_5 &= \text{abs}(x(i-1, j+1) - x(i+1, j-1)) \end{aligned} \quad (6)$$

Then we calculate the correlation value (*cor*) of five directions as in Eq. (7).

$$\begin{aligned} \text{cor}_1 &= V(j-1)/\text{diff}_1 \\ \text{cor}_2 &= V(b)/\text{diff}_2 \\ \text{cor}_3 &= V(j)/\text{diff}_3 \\ \text{cor}_4 &= V(c)/\text{diff}_4 \\ \text{cor}_5 &= V(j+1)/\text{diff}_5 \end{aligned} \quad (7)$$

Third, we calculate weight function as in Eq. (8) to solve the problem about more than one dominated edge existed.

$$\begin{aligned} \text{weight}_k &= \text{cor}_k / (\text{cor}_1 + \text{cor}_2 + \text{cor}_3 + \text{cor}_4 + \text{cor}_5) \\ k &= 1, 2, 3, 4, 5 \end{aligned} \quad (8)$$

Fourth, we calculate average value of five directions as in Eq. (9).

$$\begin{aligned} \text{average1} &= (x(i-1, j-1) + x(i+1, j+1))/2 \\ \text{average2} &= (b + g)/2 \\ \text{average3} &= (x(i-1, j) + x(i+1, j))/2 \end{aligned} \quad (9)$$

$$\text{average4} = (c + f)/2$$

$$\text{average5} = (x(i-1, j+1) + x(i+1, j-1))/2$$

As the result, we can finally interpolate the value $x(i, j)$ by Eq. (10).

$$x(i, j) = \sum_{k=1}^5 \text{weight}_k \times \text{average}_k \quad (10)$$

3.3 Block Type Decision

Block type decision is used to replace motion detection of MA method. It can help us to select the most adaptive interpolated method for each block, and avoid the computation on MC method for non-adaptive area. It consists of scene change detection, field boundary detection, motion detection, and edge detection. Figure 6 shows the block diagram of block type decision. For the scene change detection, this step will detect both of the current top field and current bottom field having scene change or not. If scene change is happened, our EAWA is applied. For the field boundary detection, it will detect the area which is belonged to field boundary or not. We use a border as the field boundary and certainly a border is a small area of the overall field. In our design the width of the border is set as one-tenth of the width of the field. For detected boundary area, we use PFI method as interpolation. Referring to motion detection, we can use it to easily find the pixel in stationary area or background area. Because the scan lines of top and bottom field are different, motion detection from interlaced video sequence needs the special treatment. We calculate the same field (even-even, odd-odd) difference to distinguish whether the block is moving or not. For blocks which are treated as moving area, it will go to the next following stage otherwise it will be treated as stationary area and interpolated by PFI method. The final stage, the Sobel edge detection, is the most complicated module in our block type decision. We perform it to

Fig. 6 Block diagram of block type decision.

Fig. 7 Flow chart of Sobel edge detection.

Fig. 8 The first step: Gauss smoothing.

detect whether the block is belonged to texture area or not. It will detect every pixel of the block with edge existence. After the detection, we will count how many pixels are existed in edge, and decide whether the block is belonged to the texture area or not. For texture area, EAWA is used. All the left blocks then finally go to the two-field MC method.

Since the Sobel edge detection is the most complicated module in our block type decision, we depict the flow chart of Sobel edge detection in Fig. 7. In Sobel edge detection, the first step is Gauss smoothing processing which is chosen to reduce the noise of the video sequence. As shown in Fig. 8, we use 5×5 Gauss smoothing mask to get Gauss smoothing value. Because of the 5×5 size of the mask, we must extend the block to 12×12 for boundary pixel.

In the second step, we use Sobel edge mask to get the horizontal and vertical edge coefficient of every pixel. Fig. 9(a) and (b) show the Sobel horizontal and vertical mask. In the third step, we utilize the edge coefficient to get the edge threshold value. We adopt edge threshold which is

(a)

(b)

Fig. 9 (a) The second step: Sobel horizontal edge detection, and (b) the Sobel vertical edge detection.

twice of the mean of the vertical/horizontal intensity in the entire block, as shown in Eq. (11). S_v and S_h are the vertical and horizontal edge value of the pixel.

$$th_v = 2 * \sum_{i=1}^{block_size} \sum_{j=1}^{block_size} S_v(i, j) / block_size^2$$

$$th_h = 2 * \sum_{i=1}^{block_size} \sum_{j=1}^{block_size} S_h(i, j) / block_size^2 \quad (11)$$

With the edge threshold, we can detect the pixel of edge existence. If one of vertical and horizontal edge values is bigger than the threshold, we will conclude the pixel has edge existence. Otherwise, we will conclude that the pixel has no edge existence. Finally, we count how many pixels will be distinguished as edge pixels as in Eq. (12). If $edge_num$ is bigger than $edge_block$ threshold value, we

will treat the block as texture area, and avoid the interpolation with MC method. Here $edge_block$ threshold is set between 10 and 30.

$$\begin{aligned} & \text{if } (S_v(i, j) > th_v \parallel S_h(i, j) > th_h) \\ & \quad edge_num++; \\ & \text{else} \\ & \quad edge_num = edge_num; \end{aligned} \quad (12)$$

4. Experiment Results and Analysis

4.1 Objective and Subjective Analysis

The detailed specification of our MC method includes the 8×8 block size and the 16×16 search range. The similarity measure is based on the common PSNR criteria. Full search algorithm is used. Of course, for a fair comparison, ours and the other MC-based methods are constructed with the same implementation specification above. We have made several common test sequences for experiment, such as *Silent*, *Weather* and *Mobile*. These sequences are widely chosen in previous literatures. Moreover, we provide the movie video as experiment. Figure 10 and Fig. 11 show the video resulted from the action movie of *Fearless*. As the objective evaluation in Fig. 10, the comparison of peak signal-to-noise ratios (PSNR) reveals that our algorithm has higher quality than MC method. This is because the blind motion estimation/compensation throughout the image is not well facilitated. The content adaptive issue is really needed for high performance concern. A more comprehensive comparison is illustrated in Fig. 11. The blurred effects are occurred by using MC method to interpolate the value. As in Fig. 11 (c), we can refine the blurred effects on field boundary and object boundary in the right and left side objects to a better video quality.

4.2 Computation Complexity Analysis

Not only reflect the video quality analysis, we also make the computation complexity analysis. Our experimental environment is on AMD 64 bit 3000 CPU processor, 1 G DDR RAM. Figure 12 shows the percentage of computation complexity, where “Others” part includes data transfer, divide field into the blocks, and if-else condition judgment. In our analysis, we observe that the two-field MC method, which intuitively is a motion estimation engine, takes the majority of total computation amount. Compared to two-field MC method, block type decision needs fewer computation powers. If we can use an accurate block type decision to avoid the area which is not adaptive to be interpolated by two-field MC method, we can save lots of computation power and even get better video quality. The CPU time of our proposed algorithm is 0.11 second per frame. If we use traditional MC method, it needs 0.578 second per frame. Compared to MC method, the CPU time of our proposed algorithm is one-fifth of it.

Fig. 10 The PSNR comparison of MC method and the proposed algorithm.

Fig. 11 The comparison of the subject view for the (a) original frame (b) MC method (c) our proposed algorithm.

Table 1 shows the number of operation on the block type decision and MC method. The block type decision obviously has the lower computation operation, but it contains some multiplication operation in Sobel edge detection. A more detailed profiling on Sobel edge detection is illustrated in Table 2. It shows that Gauss smoothing is

Fig. 12 Percentage of the computation complexity.

Table 1 The number of operation for the block type decision and motion compensation method.

Operations Methods		Add/Sub	Mul/Div	Shift	Compare	Total
Proposed Algorithm	Scene Change Detection	63	0	0	0	4133
	Field Boundary Detection	0	0	0	1	
	Motion Detection	32	0	0	1	
	Sobel Edge Detection	2304	836	768	128	
MC Method		35345	0	0	560	35905

Table 2 The computation profile of sobel edge detection.

	Add/Sub	Mul/Div	Shift	Compare
Gauss Smoothing	1536	832	512	0
Vertical Mask	320	0	128	0
Horizontal Mask	320	0	128	0
Determining Threshold	128	4	0	0
Edge Detection	0	0	0	128
Total	2304	836	768	128

the computation-intensive function, compared to other functions. In fact, in view of programming-based processor, multiplication operation can be solved as a single instruction cycle easily.

5. DSP Platform Realization

5.1 Features on DSP Platform

In order to demonstrate the proposed algorithm and achieve the real-time request, we implement our de-interlacing system on ATEME IEKC64x platform. This platform is an evaluation and development board designed with Texas Instruments DSP TMS320C6416. The processor of the platform is C6416 which is 8-way Very Long Instruction Word (VLIW) processor core. C6416 consists of eight functional units, two register files, and two data paths. It can process

Fig. 13 Implementation flow chart for ATEME IEKC64x platform.

Fig. 14 ATEME IEKC64x simulation environment.

two 16-bit $\times$ 16-bit multiplies in every cycle. Maximum frequency of C6416 is 600MHz. C6416 also has two way caches memory that can operate at the full speed of processor access. The platform also supports PAL and NTSC video capture and generation. A/V interface I/O can capture QCIF (176×144), CIF (352×288), and FULL (720×480) frame size by composite video (CVBS) or separated video (Y/C). Regardless of frame size, SVGA output can expend the frame size of 800×600 to display.

5.2 Implementation and Simulation Environment

Figure 13 shows the implementation flow chat of ATEME IKC64x platform. We capture the video frame in the beginning. Because scene change detection needs the frame information, we perform scene change detection after capturing the frame. Then we perform the proposed de-interlacing algorithm. Since the processing unit of two-field MC method and block type decision is 8×8 block, we divide the field into blocks. After dividing frame into blocks, we make block type decision to select the most adaptive method to interpolate it. The whole development and simulation system is shown in Fig. 14. We use DVB set top box (STB) as

Table 3 Code size analysis of ATEME IEKC64x.

Function	Code Size
EAWA	1428
Two-Field Motion Compensation Method	1524
Motion Detection	228
Edge Detection	1484
Scene Change Detection	652

Table 4 The run time comparison on DSP platform.

Methods Sequence	MC Method (ms/frame)	Our Proposed Algorithm (ms/frame)	Saving Computation Amount (%)
<i>Silent</i>	101.38	11.81	88.35
<i>Weather</i>	101.38	5.39	94.68
<i>Mobile</i>	101.38	45.78	54.84
<i>Mother Daughter</i>	101.38	4.05	96.00
<i>Stefan</i>	101.38	41.80	58.76
<i>Hall Monitor</i>	101.38	4.25	95.80
<i>Table Tennis</i>	101.38	45.77	54.85
<i>Coastguard</i>	101.38	34.08	66.38

input signal. By using A/V interface, we can catch the CIF video frames and store these frames into SDRAM. When the proposed algorithm is running, we can access data from SDRAM. The results show that we can real-time get the de-interlaced frames to display on the LCD monitor.

In DSP platform-based design, there are several methods to speedup our proposed algorithm. Here we use four techniques to achieve it. First is to use compiler options. Compiler options can speedup or reduce code size. Second is to eliminate redundant loops. We can reduce two dimension loop to one dimension, or parallel process multiply instruction. Third is the memory allocation. For embedded system, memory is limited and critical to the system performance. Fourth is to define some critical thresholds as constant.

Table 3 shows code size of ATEME IEKC64x. The memory usage is proportional to the code size. Because two-field motion compensation method needs two frame buffers, its code size is the largest. Motion detection only needs two block buffers, so its code size is the smallest. The platform-based run time analysis and comparison with the proposed method and MC method are illustrated in Table 4. Since a blind processing on motion estimation/compensation is applied in MC method, the processing time is fixed as 101.38 ms per frame. Our de-interlacing algorithm is content-dependent. Briefly, we achieve a low computation power design. The saving ratio is about 54% to 96%.

6. Conclusions

Since few literatures explored the de-interlacing design including both the algorithm and hardware level perspectives, in this paper a content adaptive de-interlacing technique is presented in detail. In our algorithm, it includes several efficient techniques, especially for EAWA and block type decision. EAWA is a novel intra-field method. Compared with traditionally ELA, we can detect the edge more accurately, and thus it is more adaptive for texture area. Block type decision is the other important idea in our proposed algorithm. With this mechanism, we can precisely classify the inter- or intra-field method without the blind motion estimation on every pixel. This paper presents a comprehensive description on the proposed algorithm and also provides complete numerical data for comparisons. Not only the subjective view, but also the objective evaluation is provided. Experimental results show that we can save large computational complexity and also get higher video quality than other methods. Referring to the design on platform-based system, we apply several optimization techniques to enhance the system performance.

References

- [1] G. De Haan and E.B. Bellers, "De-interlacing—An overview," *Proc. IEEE*, vol.86, no.9, pp.1839–1855, Sept. 1998.
- [2] H.Y. Lee, J.W. Park, T.M. Bae, S.U. Choi, and Y.H. Ha, "Adaptive scan rate up-conversion system based on human visual characteristics," *IEEE Trans. Consum. Electron.*, vol.46, no.4, pp.999–1005, Nov. 2000.
- [3] Y.-L. Chang, S.-F. Lin, and L.-G. Chen, "Extended intelligent edge-based line average with its implementation and test method," *Proc. 2004 Int. Symp. Circuits and Systems (ISCAS 2004)*, pp.341–344, Vancouver, Canada, May 2004.
- [4] H. Yoo and J. Jeong, "Direction-oriented interpolation and its application to de-interlacing," *IEEE Trans. Consum. Electron.*, vol.48, no.4, pp.954–962, Nov. 2002.
- [5] H.-H. Wu, M.-H. Sheu, and T.-Y. Yang, "Directional interpolation for field-Sequential stereoscopic video," *ISCAS 2005*, 2005.
- [6] M.K. Park, M. Byun, J.H. Park, and M.G. Kang, "Edge-dependent interpolation-based deinterlacing using edge patterns," *Journal of Electronic Imaging*, vol.15, 043003, Dec. 2006.
- [7] K. Sugiyama and H. Nakamura, "A method of de-interlacing with motion compensated interpolation," *IEEE Trans. Consum. Electron.*, vol.45, no.3, pp.611–616, Aug. 1999.
- [8] O. Kwon, K. Sohn, and C. Lee, "Deinterlacing using directional interpolation and motion compensation," *IEEE Trans. Consum. Electron.*, vol.49, no.1, pp.198–203, Feb. 2003.
- [9] Y.-L. Chang, P.-H. Wu, S.-F. Lin, and L.-G. Chen, "Four field local motion compensated de-interlacing," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP 2004)*, pp.253–256, Montreal, Canada, May 2004.
- [10] S.-F. Lin, Y.-L. Chang, and L.-G. Chen, "Motion adaptive interpolation with horizontal motion detection for deinterlacing," *IEEE Trans. Consum. Electron.*, vol.49, no.4, pp.1256–1265, Nov. 2003.
- [11] M.-J. Chen, C.-H. Huang, and C.-T. Hsu, "Efficient de-interlacing technique by inter-field information," *IEEE Trans. Consum. Electron.*, vol.50, no.4, pp.1202–1208, Nov. 2004.
- [12] G.-L. Li and M.-J. Chen, "High performance de-interlacing algorithm for digital television displays," *IEEE Journal of Display Technology*, vol.2, no.1, pp.85–90, March 2006.

- [13] D. Wang, A. Vincent, and P. Blanchfield, "Hybrid de-interlacing algorithm based on motion vector reliability," *IEEE Trans. Circuits Syst. Video Technol.*, vol.15, no.8, pp.1019–1025, Aug. 2005.
- [14] X. Gao, J. Gu, and J. Li, "De-interlacing algorithms based on motion compensation," *IEEE Trans. Consum. Electron.*, vol.51, no.2, pp.589–599, May 2005.
- [15] C.C. Lin, M.H. Sheu, H.K. Chiang, and C.J. Wei, "The VLSI design of motion adaptive de-interlacing with horizontal and vertical motions detection," *APCCAS 2006*, pp.1587–1590, Dec. 2006.
- [16] C.C. Lin, C.J. Wei, M.H. Sheu, H.K. Chiang, and C. Liaw, "The VLSI design of de-interlacing with scene change detection," *ISCAS 2006*, May 2006.
- [17] R. Li, B. Zeng, and M.L. Liou, "Reliable motion detection/compensation for interlaced sequences and its application to deinterlacing," *IEEE Trans. Circuits Syst. Video Technol.*, vol.10, no.1, pp.23–29, Feb. 2000.

Tsung-Han Tsai received the B.S., M.S., and Ph.D. degrees in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 1990, 1994, and 1998 respectively. Dr. Tsai was an Instructor (1994–1998) and an Associate Professor (1998–1999) of the department of electrical engineering at Hwa Hsia College of Technology and Commerce. From 1999 to 2000, he was an Associate Professor of electronic engineering at Fu Jen University. In 2000, he joined the Department of Electrical Engineering,

National Central University, Taiwan, where he is currently an Associate Professor. He has been an IEEE member for over 10 years, and is also a member of the Audio Engineering Society (AES). Dr. Tsai has been awarded 10 patents and has published more than 100 referred papers in international journals and conferences. His research interests include VLSI signal processing, video/audio coding algorithms, DSP architecture design, wireless communication, and System-On-Chip design. Dr. Tsai received the Industrial Cooperation Award in 2003 from the Ministry of Education, Taiwan. He is a member of the Technical Committee of IEEE Circuits and Systems Society, and serves as Technical Program Committee member or Session Chair of several international conferences.

Hsueh-Liang Lin received the B.S. degrees in electrical engineering from the National Central University, Taiwan, in 2006. His areas of interests are video signal processing and VLSI signal processing.