Hindawi Publishing Corporation
EURASIP Journal on Image and Video Processing
Volume 2008, Article ID 741290, 10 pages
doi:10.1155/2008/741290

Research Article

A Motion-Adaptive Deinterlacer via Hybrid Motion Detection and Edge-Pattern Recognition

Gwo Giun Lee,[1] Ming-Jiun Wang,[1] Hsin-Te Li,[2] and He-Yuan Lin[1]

[1] *Department of Electrical Engineering, National Cheng Kung University, 1 Ta-Hsueh Road, Tainan 701, Taiwan*
[2] *Sunplus Technology Company Ltd, 19 Chuangsin 1st Road, Hsinchu 300, Taiwan*

Correspondence should be addressed to Ming-Jiun Wang, n2894155@ccmail.ncku.edu.tw

Received 31 March 2007; Revised 25 August 2007; Accepted 13 January 2008

Recommended by J. Konrad

A novel motion-adaptive deinterlacing algorithm with edge-pattern recognition and hybrid motion detection is introduced. The great variety of video contents makes the processing of assorted motion, edges, textures, and the combination of them very difficult with a single algorithm. The edge-pattern recognition algorithm introduced in this paper exhibits the flexibility in processing both textures and edges which need to be separately accomplished by line average and edge-based line average before. Moreover, predicting the neighboring pixels for pattern analysis and interpolation further enhances the adaptability of the edge-pattern recognition unit when motion detection is incorporated. Our hybrid motion detection features accurate detection of fast and slow motion in interlaced video and also the motion with edges. Using only three fields for detection also renders higher temporal correlation for interpolation. The better performance of our deinterlacing algorithm with higher content-adaptability and less memory cost than the state-of-the-art 4-field motion detection algorithms can be seen from the subjective and objective experimental results of the CIF and PAL video sequences.

1. INTRODUCTION

Interlaced scanning, or interlacing, which performs vertical-temporal subsampling of video sequences was used to lower the costs of video system and reduce transmission bandwidth by half while retaining visual quality in traditional TV. One common characteristic of many television standards evolving over time, such as PAL, NTSC, and SECAM, is interlaced scanning. With recent advancements of digital TV (DTV), high-definition TV (HDTV), and multimedia personal computers, deinterlacing has become an important technique which converts interlaced TV sequences into frames for display on progressive devices such as LCD TVs, plasma display panels, and projective TVs. Intrinsic to this interoperability of the two seemingly separate domains is the conversion of interlaced TV formats to progressive displays via deinterlacers. Hence, the increased demand for research in video processing systems to produce progressively scanned video with high-visual quality is inevitable [1].

The deinterlacing problem can be stated as

$$p_o(i,j,k) = \begin{cases} p_i(i,j,k), & (j+k)\%2 = 0, \\ \hat{p}(i,j,k), & \text{otherwise}, \end{cases} \tag{1}$$

where p_i, $\hat{p}$, and p_o denote the input, interpolated, and output pixels, respectively. i, j, and k represent horizontal, vertical, and temporal pixel indices. % is modulo operation. The vertical-temporal downsampling structure of interlacing is also explained in (1), in which $\hat{p}$ indicates the missing point due to interlacing.

The challenge of deinterlacing is to interpolate the missing points $\hat{p}$ with limited information and also to maintain clear visual quality as well. However, visual defects such as edge flicker, line crawling, blur, and jaggedness due to the inherent nature of interlaced sequences frequently appear and produce annoying artifacts to viewers if deinterlacing is not done properly.

The key concept of deinterlacing is to interpolate the missing point with spatio-temporal neighbors that have the highest correlation. A wide variety of deinterlacing algorithms, following this principle, has been proposed in the last few decades. A comprehensive survey can be found in [2]. We introduce several frequently used techniques, which are helpful in understanding this paper, in the following.

Since $p_i(i,j-1,k)$, $p_i(i,j+1,k)$, and $p_i(i,j,k-1)$ are the nearest neighbors of $\hat{p}(i,j,k)$, they potentially have the highest correlation. Two simple interpolation strategies, line average (LA) and field insertion (FI), were hence proposed. LA, an intra-interpolation method, interpolates $\hat{p}(i,j,k)$ with $(p_i(i,j-1,k),\ p_i(i,j+1,k))/2$. On the other hand, FI, an inter-interpolation method, repeats $p_i(i,j,k-1)$ as $\hat{p}(i,j,k)$. LA and FI methods are so simple that they cannot handle generic video contents. LA blurs vertical details and causes temporal flickering. FI introduces line crawl of moving objects.

Another intra-interpolation method, called edge-based line average (ELA) [3], was proposed to preserve edge sharpness and integrity and avoid jaggedness of edges. ELA interpolates a pixel along the edge direction explored by comparing the gradients of various possible directions. Although ELA is capable of restoring the edges of interlaced video, it also introduces "pepper & salt" noises when edge directions are misjudged. Moreover, the weakness in recovering complex textures is one of its drawbacks. Some variations of ELA, such as adaptive ELA [4], enhanced ELA (EELA) [5], and extended intelligent ELA algorithm [6] were proposed to further improve its performance.

Motion-adaptive methods [7–9] were proposed to alleviate the impact of motion so that the correlation of the reference pixels for interpolation is higher. Motion-adaptive deinterlacing employs motion detection (MD), and switches or fades between filtering strategies for motion and non-motion cases by calculating the differences of luminance between several consecutive fields. A good survey on motion detection of interlaced video can be found in [10]. Motion detection requires field memories to store previous fields and possibly future fields. With more fields and thus more information, the detection accuracy is usually higher at the cost of more field memory in VLSI implementation. In motion-adaptive deinterlacing, intra-interpolation is selected for motion cases, while inter-interpolation is used for stationary scenes. The visual quality of motion-adaptive methods highly relies on the correctness of motion information. Textures make correct motion detection, especially the detection of fast motion, even more difficult, since the vertical and temporal high frequencies are mixed up in interlaced video. It was reported in [11] that texture analysis by wavelet decomposition can enhance the precision of motion detection. Motion-compensated methods [12–15] involve motion estimation [16–18] for filtering along the motion trajectories. They perform very accurate interpolation at the cost of much higher hardware expenditure.

Rich video contents provide viewers with high-visual satisfaction but complicate the deinterlacing process, since different visual signal processing strategies should be applied to the video signals with more information. The motion-adaptive method with FI and LA switches between a vertical all-pass filter and a temporal all-pass filter and hence provides a content-adaptive algorithm. However, the adaptability of motion detection and intra-interpolation did not draw much attention before. The earlier motion detection algorithms focused on accurate same-parity detection but neglected the detection of fast motion. Moreover, increasing the number of fields to obtain higher accuracy also accompanies higher cost of memory hardware. On the other hand, ELA-styled interpolations emphasized the sharpness of

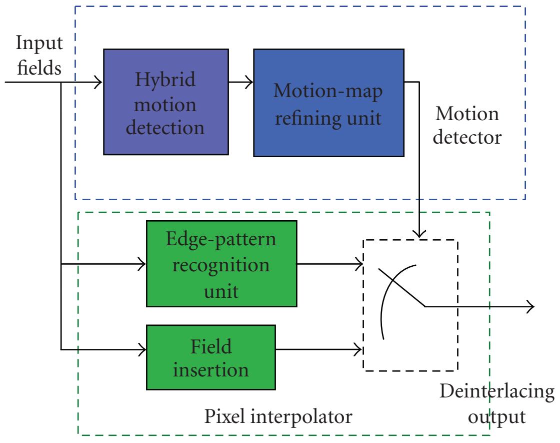


Figure 1: Block diagram of the proposed deinterlacing algorithm.

edges but ignored the importance of textures. The robustness of these algorithms toward different video contents can still be enhanced.

In this paper, we present a hybrid motion-adaptive deinterlacing algorithm (HMDEPR), which consists of novel hybrid motion detection (HMD) and edge-pattern recognition (EPR) with emphasis on content-adaptive processing. HMD is capable of detecting versatile motion scenarios by using only three fields. EPR targets the interpolation of edges and textures, which can not be handled by using either LA or ELA alone. The experimental results indicate that our HMD, EPR for intra-interpolation, and our deinterlacing algorithm all exhibit higher robustness toward assorted video scenes. This paper is organized as follows. Section 2 presents our motion-adaptive deinterlacing algorithm. The experimental results and performance comparison are shown in Section 3. The conclusion of this research is drawn in Section 4.

2. THE PROPOSED DEINTERLACING ALGORITHM

We introduce a deinterlacing algorithm which adapts to the motion, texture, and edge contents of the video sequence. The overall algorithm, shown in Figure 1, consists of a motion detector and a pixel interpolator. Our motion detector employs HMD and a refinement unit. The interpolator includes EPR and FI. FI is used when a pixel is detected as stationary, and EPR is used otherwise. HMD and EPR are tactically designed to achieve high adaptability towards a great variety of motion, textures, and edges.

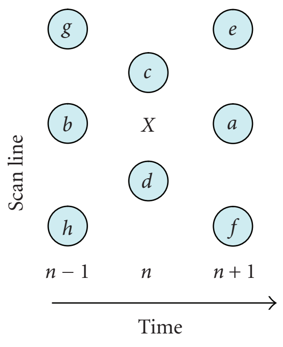
2.1. *Motion detection*

The goal of motion detection is to identify motion scenes and enable intra-interpolation. We employ a hybrid motion detector (HMD) which requires the pixel data of only three fields. The pseudo-codes of the HMD are shown in Figure 2. The three conditions are dedicated to the detection of slow motion, fast motion, and motion with edges.

The first condition of HMD is traditional 3-field motion detection. The 3-field motion detection is capable of detecting most of the motion scenarios except the case in which

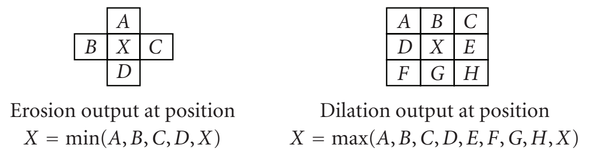
```
$\text{diff1} = \text{abs}(a - b)$
$\text{diff2} = \text{abs}[b - (c + d)/2]$
$\text{diff3} = \text{abs}[b - (g + h)/2]$
$\text{diff4} = \text{abs}[a + (e + f)/2 - b - (g + h)/2]$

if diff1 > TH1                               ▷ 1st condition
  flag ⟵ motion
else if diff2 > TH1 AND diff3 < TH2          ▷ 2nd condition
  flag ⟵ motion
else if diff4 > TH3                          ▷ 3rd condition
  flag ⟵ motion
else
  flag ⟵ stationary
```

(a)



(b)

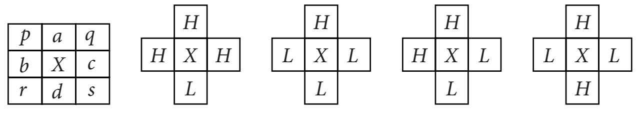
Figure 2: The proposed hybrid motion detection algorithm. (a) Pseudocodes, (b) pixel definition.


A
B X C
D
Erosion output at position
X = min(A, B, C, D, X)
A B C
D X E
F G H
Dilation output at position
X = max(A, B, C, D, E, F, G, H, X)

(a) Erosion output at position X = minimum (A, B, C, D, X)

(b) Dilation output at position X = maximum (A, B, C, D, E, F, G, H, X)

Figure 3: Nonsymmetric opening operation. (a) Erosion, (b) dilation.



(a) (b) (c) (d) (e)

Figure 4: Edge pattern. (a) Pixel definition, (b) $3H1L$ pattern, (c) $3L1H$ pattern, (d) $2H2L$ corner pattern, (e) $2H2L$ stripe pattern.

moving objects or backgrounds only appear in field n but neither in field $n - 1$ nor field $n + 1$. This is supported by the fact that near-zero difference of $a - b$ in this case falsely indicates no motion. Hence, in the second condition, we further take the line average result as a temporarily interpolated point and detect motion between two consecutive fields under the condition that the vertical variation *diff3* is small. This 2-field motion detection operates on the previous field rather than the next field so as to coherently work with FI from the previous field in stationary scenes. The proposed HMD combines the merits of 3-field motion detection and 2-field motion detection, which are good detection accuracy for stationary pixels, and the ability to detect the very fast motion that cannot be detected by 3-field motion detection, respectively. The third condition enhances the detection accuracy of edges. The reason will be explained in Section 3.2 with motion maps.

We employ a binary nonsymmetric opening morphological filter to further refine the motion map of HMD. First, an erosion filter with a cross-shaped mask, as shown in Figure 3(a), is performed on the results of HMD. The erosion filter eliminates isolated moving pixels. A dilation filter with a 3×3 mask, as shown in Figure 3(b), is performed after erosion to restore and extend the shape of moving objects after erosion. The inability to detect motion is referred to as *motion missing,* which results in *motion holes* on the interpolated images, and the detected motion of stationary objects as *false motion*. The nonsymmetric opening morphological filter can minimize the visual artifact caused by motion missing problem and enhance the overall performance in spite of the corresponding false motion problem.

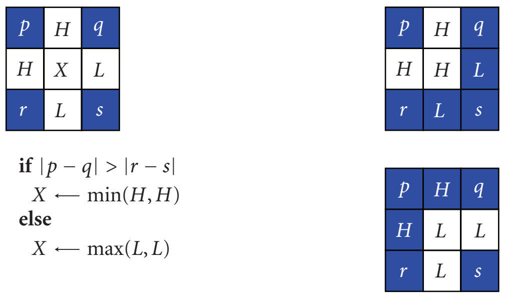
In our HMD, two filtering strategies, that is, the first and the second conditions, are used to cover motion with various speed and result in correct detection of motion. Moving edges are also considered by the third condition. The inclusive detection strategies contribute to the adaptability of HMD. Moreover, the short field delay offers higher temporal correlation for interpolation than 4-fileld and 5-field algorithms. Its low-memory cost also makes it attractive for VLSI implementation.

2.2. Interpolation

There are two interpolation schemes in the proposed motion-adaptive deinterlacing algorithm to be chosen from. EPR is the intra-field interpolator used in moving scenes, and FI is the inter-field interpolator used in stationary scenes. HMD adaptively selects intra-field or inter-field interpolation as the output.

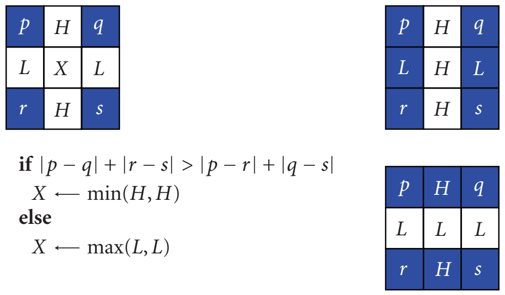
Moving textures are very difficult to interpolate because aliasing may already exist after interlacing as explained by the spectral analysis in [2]. Inspired by color filter array, which has been widely used in cost-effective consumer digital still cameras [19], we adapt it for texture and edge interpolations. In Figure 4, there are four unique types of edge patterns within a 3×3 window, which are $3H1L$ edge patterns, $3L1H$ edge patterns, $2H2L$ corner patterns, and $2H2L$ stripe patterns. The definition of "H" and "L" pixels is similar to delta modulation in communications systems. If the pixel value is larger than the average of pixel a, b, c, and d, it is marked as "H" and marked as "L" otherwise. We can obtain 14 distinct patterns with different orientations.

By considering $3H1L$ and $3L1H$ edge patterns in Figures 4(b) and 4(c), it is obvious that the center pixel X is very likely to be one of the majority neighbor pixels. Hence, the



```
if $|p-q| > |r-s|$
    $X \longleftarrow \min(H,H)$
else
    $X \longleftarrow \max(L,L)$
```

Figure 5: The interpolation method of $2H2L$ corner pattern.



```
if $|p-q| + |r-s| > |p-r| + |q-s|$
    $X \longleftarrow \min(H,H)$
else
    $X \longleftarrow \max(L,L)$
```

Figure 6: The interpolation method of $2H2L$ stripe pattern.

median of H or L pixels around X are calculated as the interpolation result. Consider the $2H2L$ corner pattern shown in Figure 4(d), the gradients in horizontal directions, $|p-q|$ and $|r-s|$, are computed. If $|p-q|$ is larger than $|r-s|$, it indicates that the gradient at upper position is larger than that at lower position. There is an "H" corner in the 3×3 window. In this condition, the function minimum(H, H) is applied to interpolate pixel X; Otherwise, there is a "L" corner and the function maximum(L, L) is applied as shown in Figure 5. The other three corner patterns can be calculated in a similar way. As for the $2H2L$ stripe pattern in Figure 4(e), four gradient values, $|p-q|$, $|r-s|$, $|p-r|$, and $|q-s|$ are computed here. As shown in Figure 6, if the sum of horizontal gradient values is larger than the sum of vertical ones, minimum(H, H) is applied because there exists a vertical edge; otherwise, the function maximum(L, L) is used due to a horizontal edge. The median filter of $3H1L$ and $3L1H$ patterns, and the minimum filter and maximum filter of "H" pixels and "L" pixels can avoid the interpolation of an extreme value and thus minimize the risk of "pepper & salt" noises.

The pixels b and c in Figure 4(a), like X, are also missing pixels in interlaced video. To prevent error propagation, we adaptively obtain b from the previous field if it is detected as stationary and from the average of p and r if it is moving. Likewise, the value of c can be calculated for EPR of X. Predicting b and c adaptively from either spatial or temporal neighbors greatly increases their correlation with the X, which again helps with the pattern analysis and interpolation.

EPR provides a low-complexity deinterlacer which efficiently adapts to textural and edge contents during interpolation. The uniqueness in using EPR in deinterlacing is that complex scenes with textures or edges are analyzed and categorized into reasonable number of patterns by delta modulation, which adaptively determines on the prediction value for the one-bit encoding of the four pixels and thus accommodates extensive cases of input video. The pattern encoding is followed by an associated filtering scheme using the contextual information from the vicinity having high correlation. The simple hardware realization of delta modulation and the corresponding operations also makes EPR more favorable.

Table 1: The abbreviations of the algorithms.

Abbreviation	Full name
FI	Field-insertion
LA	Line-average
ELA	Edge-based line-average [3]
EELA	Enhanced edge-based line-average [5]
EPR	Edge-pattern recognition (proposed)
HMD	Hybrid motion detection (proposed)
HMDEPR	Motion-adaptive deinterlacing with HMD and EPR (proposed)
2FMA	2-field motion-adaptive deinterlacing
3FMA	3-field motion-adaptive deinterlacing
4FTD	4-field motion-adaptive algorithm with texture detection [21]
4FHMD	4-field motion-adaptive algorithm with horizontal motion detection [5]

3. EXPERIMENTS AND THE RESULTS

Deinterlacing is commonly applied to standard definition (SD) video signals such as PAL and NTSC. We have experimented on several SD video sequences for subjective comparison. However, to objectively and comprehensively present the performance of HMDEPR, we also show the results of CIF video sequences. When the same continuous video signal is sampled in CIF and SD resolution, the CIF sequence would have a wider spectrum and have more high-frequency components near the interlace replicas than SD after interlacing [2, 20]. The CIF sequences are thus used as critical test conditions.

We compared our algorithm to LA, FI, 2-field motion-adaptive algorithm (2FMA), 3-field motion-adaptive algorithm (3FMA), 4-field motion-adaptive algorithm with texture detection (4FTD) [21], and 4-field motion-adaptive algorithm with horizontal motion detection (4FHMD) [5]. To facilitate the reading, we summarize the abbreviation of these algorithms in Table 1. The detailed setting of all algorithms and other experimental conditions are described in Section 3.1. To clearly demonstrate the performance of our algorithm, we separate the experiments into three parts. Section 3.2 analyzes our motion-detection algorithm with the contribution of each step. The second part, shown in Section 3.3, compares the EPR algorithm with other intra-interpolation methods. In Section 3.4, we combine FI, EPR,

```
if abs($p_i(i, j-1, k) - p_i(i, j, k-1)$) > $TH_{2FMD}$
    line-average,
else
    field-insertion,
```

Algorithm 1

```
if abs($p_i(i, j, k-1) - p_i(i, j, k+1)$) > $TH_{2FMD}$
    line-average,
else
    field-insertion.
```

Algorithm 2

```
if max_diff ≥ $TH_{4FTD\_Motion}$,
        AND Var > $TH_{4FTD\_Texture}$
    3D − ELA,
else if max_diff < $TH_{4FTD\_Motion}$,
        AND Var > $TH_{4FTD\_Texture}$
    modified-ELA,
else if max_diff ≥ $TH_{4FTD\_Motion}$,
        AND Var ≤ $TH_{4FTD\_Texture}$
    VT-linear-filter,
else
    VT-median-filter,
```

Algorithm 3

and HMD as the deinterlacer and show its subjective and objective performance and comparison.

3.1. Experimental settings

2FMA and 3FMA are two simple motion-adaptive algorithms used to highlight the accuracy of HMDEPR. The detail of 2FMA is described as in Algorithm 1 where the symbolic definition is the same as in 1.

3FMA, also known as the simplest same-parity motion detection, is described as in Algorithm 2.

4FTD [21] performs not only motion detection, but also texture detection. The simplified algorithm is described as in Algorithm 3 where max_diff is the maximum of three absolute differences in the 4-field motion detection. Var is the variance of the 3×3 spatial block centered at the current pixel. This algorithm classifies the current pixel as one of the four cases: moving textural region, moving smooth region, static textural region, and static smooth region with associated interpolation methods. The pixels used in 3-dimensional (3D) ELA in [21] are missing pixels. In our experiment, we reasonably use

$$\{p_i(i+m, j+n, k+l) \mid m \in \{-1,0,1\},\\ n \in \{-2,2\}, l \in \{-1,1\}\}. \tag{2}$$

4FHMD [5] performs horizontal motion detection. If the temporal difference is smaller than an adaptive threshold, temporal interpolation along the moving direction will be adopted. Otherwise, 5-tap EELA will apply to the motion scenes. In the EELA, the threshold TH_{EELA} is required to ensure a dominate edge and thus avoid “pepper & salt” noises caused by edge misjudgments. The current pixel for threshold adjustment is missing, which is not explained in [5]. We use the line average result for threshold adjustment in our implementation.

Table 2 shows the thresholds used throughout our experiments. They are tuned to achieve optimally subjective and objective output video quality. The threshold TH1 of our algorithm is set to a small value to prevent motion missing problem. Although it causes more false motion at the same time, this negative effect is alleviated by EPR. TH2 is set to prevent erroneous opposite-parity difference that appears in textural scenes. TH3 is set as the double of TH1, since two difference pairs are involved. Quantitatively, the performance is not sensitive to any thresholds in Table 2 since the PSNR difference of the test sequences is less than 0.01 dB if we increase or decrease one of the thresholds by one. Moreover, the visual difference of changing TH1 from 4 to 12 is not perceivable unless very carefully inspected.

In the determination of 2FMA, a large threshold favors stationary scenes such as Silent and Mother & daughter, while a small threshold benefits moving scenes. The value was fixed to balance the gain and loss of all sequences. Similar tradeoff was made for the other thresholds. There was, however, a special situation in determining $TH_{4FTD\_Texture}$. “Pepper & salt” noises are a serious problem in 3D ELA. Eventually, $TH_{4FTD\_Texture}$ was set to a large value to prevent choosing 3D ELA. Only sharp edges are detected as texture regions.

The PSNR of the kth frame is calculated by 5, where p_p is the pixel in progressive sequences. M and N are the frame width and frame height. Other symbolic definitions are the same as in 1. We exclude the boundary cases, the first and the last line, in PSNR calculation. The PSNR in Section 3.4 is the average value of all frames between the third frame and the $F-1$th frame, where F is the total frame number of a sequence:

$$\text{PSNR}(k) = 10\log_{10}\frac{255^2}{\left(\left(\sum_{i=0}^{M-1}\sum_{j=1}^{N-2}\left[p_o(i,j,k)-p_p(i,j,k)\right]^2\right)/MN\right)}. \tag{3}$$

3.2. Experimental results of motion detection

We take the 157th motion map of foreman in CIF resolution as an example to explain the contribution of HMD step-by-step. In Figures 7(a)–7(f), the fused 156th and 157th fields of Foreman are overlaid with motion maps. The transparent regions are the motion regions, and the opaque regions are detected as stationary. The edges of the building and the fast moving gesture are two crucial parts in motion detection, since severe line crawl will be observed if the motion is not detected correctly. All the regions with moving hand, either

Table 2: The thresholds in our experiments.

Algorithms	Thresholds	Value
HMDEPR	TH1	8
	TH2	20
	TH3	16
2FMA	TH_{2FMA}	20
3FMA	TH_{3FMA}	20
4FTD	TH_{4FTD_Motion}	80
	$TH_{4FTD\_Texture}$	80000
4FHMD	TH_{EELA}	20

Table 3: PSNR of the intra-interpolation methods in dB.

	LA	ELA [3]	EELA [5]	EPR w/o MAP	EPR with MAP
Waves	32.32	31.29	32.05	31.85	32.72
Grass	34.45	31.49	33.31	32.86	33.45
Trees	26.33	25.73	25.95	25.91	25.90
Bricks & T.	29.18	28.64	28.96	28.67	28.94
Average	30.57	29.29	30.07	29.82	30.25

in 156th or 157th fields should be detected as moving regions where FI is not applicable.

In Figure 7(a), most of the motion regions are successfully detected by the first condition. However, some gestures are still missing due to the fast movement. The second condition, targeting the detection of fast motion, solves the problem for the first condition, and the result is shown in Figure 7(b). The edges of the building are composed of two smooth regions with different luminance. Limited by the aperture problem, the motion regions of the edges detected by the first condition are very thin lines. These detected lines will be totally eroded by the morphological filter, which again causes line crawl near the edges. The second condition is not applied to the edges as a result of vertical variation checking. The third condition of HMD is therefore used to detect the motion of edges. The vertically extended detection window enlarges the detected region of edges, which will still exist after morphological operation. This is illustrated in Figure 7(c).

The motion maps of the other procedures are shown in Figures 7(d)–7(f). The initial detection result comes from the combination of the three conditions. After the erosion process, the detection noise is greatly reduced. The eroded region and some small motion holes are recovered by the dilation filter as in the final result of HMD. The final interpolation result is shown in Figure 7(g). The excellent interpolation of the edges and the fast gesture reveal the accuracy and robustness of HMD. The small motion hole of the final motion map is not obvious in the interpolated image due to the small luminance difference between the two consecutive fields. The interpolation result of 3FMA, with TH_{3FMA} being modified to eight, is shown in Figure 7(h) for comparison. The motion missing of 3FMA leads to annoying line crawling effect.

3.3. Experimental results of edge-pattern recognition

In this section, we show the comparison of five intra-interpolation methods, including LA, 5-tap ELA [3], 5-tap EELA [5], EPR without motion-adaptive prediction (MAP), and EPR with MAP. EPR with MAP adaptively predicts b and c, shown in Figure 4(a), from either LA or previous field, while EPR without MAP always determine the two pixels by LA. The EELA scheme in this section is the same algorithm used in 4FHMD with horizontal motion detection being disabled. The sequence resolution in the following two experiments is CIF.

The first experiment is to test the PSNR on different textures extracted from the video sequences. Figure 8 shows the extracted regions including waves, grass, trees, and bricks and trees. The extracted scenes are moving or partially moving, in which intra-interpolation will be adopted. The comparison is shown in Table 3. LA can be considered as a conservative method for textural scenes. ELA was primarily designed for edge interpolation. With the expectation of edge misjudgments in complex textures, the performance of ELA is the worst among all algorithms. In EELA, LA is used in textural scenes whenever the dominate edge direction cannot be found. Therefore, the performance of EELA is close to that of LA in some cases. EPR without MAP is always better than the ELA in this experiment. The performance becomes even better when motion detection is used to predict the neighboring pixels. Significant improvement can be observed in the semimotion scene like waves. The overall PSNR of EPR with MAP is better than that of EELA.

The second experiment is to test the ability of edge interpolation. In the results shown in Figure 9, interpolation with LA introduces edge jaggedness while EPR, ELA, and EELA preserve the edge sharpness. The accurate edge interpolation of EPR is contributed by its corner pattern. The neighboring pixels of the edges are detected as stationary in this example, making the interpolation result of EPR with MAP more accurate.

The performance of the intra-interpolation algorithms can be discussed in the following cases. LA has the highest PSNR in purely moving and textural scenes due to its conservative way of interpolation. With the incorporation of temporal information in semimoving scenes, our EPR performs better than LA and EELA. EPR, ELA, and EELA can all handle edges. Interpolation of textures is the weakness of ELA while edges defeat LA. EELA, which adaptively switches between ELA and LA, is capable of interpolating edges and textures. EPR has the same adaptive ability and is better than EELA due to more flexible interpolation schemes contributed by pattern analysis for complex textures and also the aid of more information provided by MAP. Hence, the content-adaptability is the advantage of EPR against LA and ELA, and EELA.

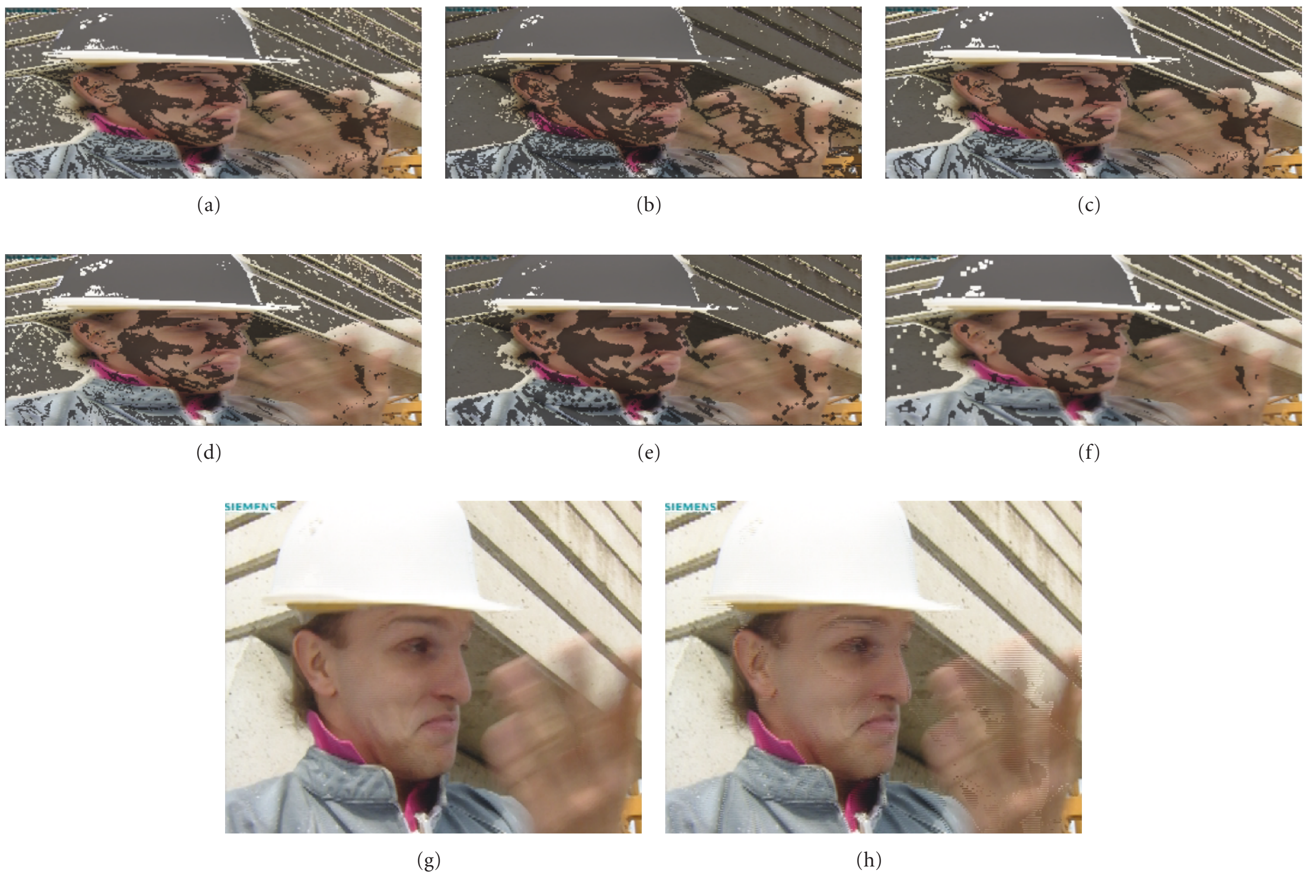


Figure 7: The results of motion detector. (a) Motion map of the 1st condition, (b) motion map of the 2nd condition, (c) motion map of the 3rd condition, (d) output of HMD, (e) eroded motion map, (f) final motion map by dilation, (g) interpolation result of the proposed algorithm, (h) interpolation result of 3-field motion detection.

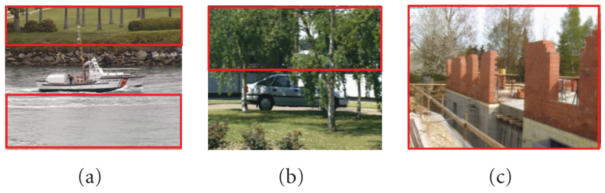


Figure 8: Extracted textures from video sequences. (a) Coastguard, 343rd picture, (b) vectra-color, 92nd picture, (c) foreman, 244th picture.

3.4. Deinterlacing with motion detection and EPR

Figures 10 and 11 show the visual quality of cropped Stefan in PAL resolution. The 50th field of Stefan contains stationary background and the moving player, which is suitable to highlight the accuracy of the different motion detectors. In particular, the fast movement of the racket, which only appears at the same position for only one picture, is very difficult to detect. In Figure 10, LA produces the blurred background and edges, and FI introduces line crawl on the player as a result of no motion-adaptive scheme. Textures in the background cause false motion detection of 2FMA, which lead to blurred vertical details. 3FMA cannot detect the movement of the player, especially the fast-moving racket, leaving many motion holes on the player. Although there is some motion missing in 4FTD, vertical-temporal median filter will adopt intra-pixels in this case and thus eliminate the motion holes. However, the regions detected as moving, such as Stefan's hands, are interpolated by vertical-temporal linear filter. Unfortunately, involving temporal pixels in moving regions still causes line crawl. 4FHMD provides a good detection result in the background. But the misjudgment of horizontal motion direction causes line crawling and also "pepper & salt" noises. HMDEPR shows accurate motion detection results in both stationary background and moving foreground, which preserves the integrity of the background textures and edges and also minimizes the motion holes of the foreground.

In Figure 11, the entire 194th field of Stefan with textures is moving and is used to compare the visual quality of textures interpolated by LA and HMDEPR. HMDEPR performs accurate motion detection as well as sharper edge and texture interpolation than LA does, and also exhibits the same quality for the textures without apparent edges.

We also present objective algorithm comparison of CIF video sequences in Table 4. The scenes with moving foregrounds and the stationary backgrounds, including

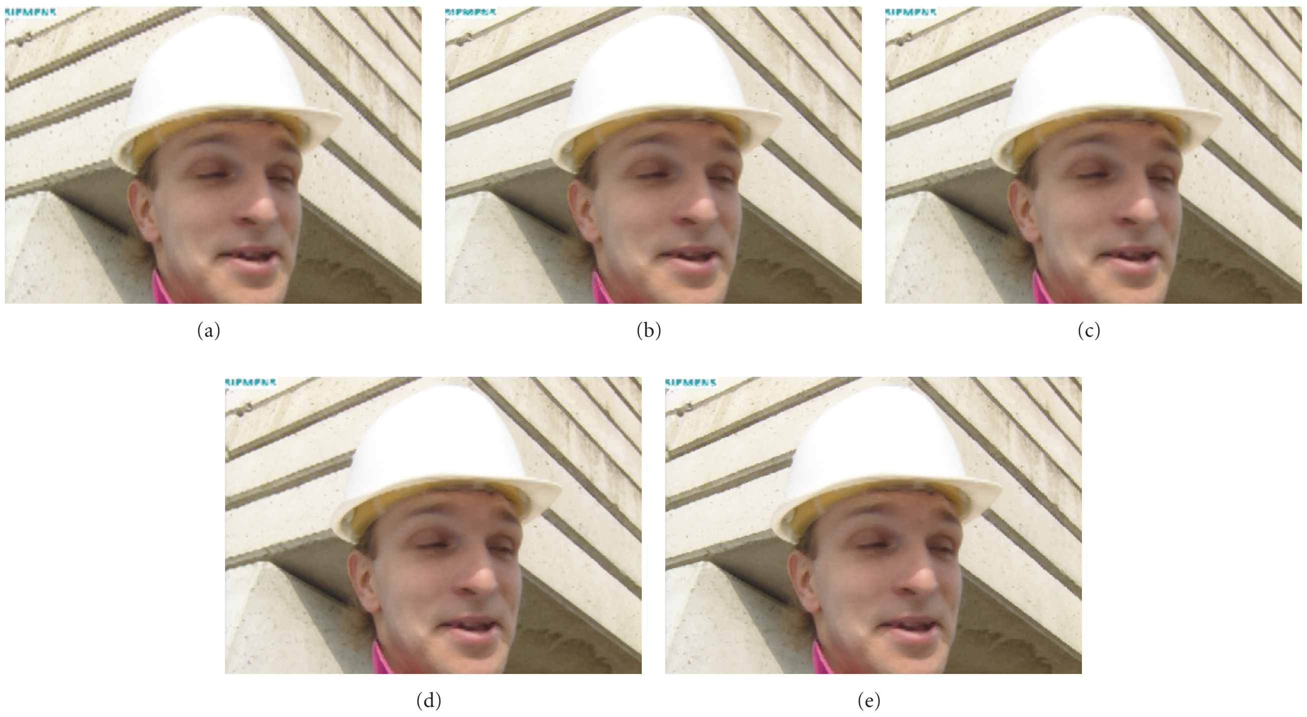


Figure 9: Visual quality of intra-interpolation methods. (a) LA, (b) ELA, (c) EELA, (d) proposed EPR without MAP, (e) proposed EPR with MAP.

Hall-monitor, Silent, and Mother & daughter, are similar to 50th field of Stefan, whose aforementioned analysis can be used to explain the PSNR's of these three sequences. In the sequence Hall-monitor, LA has the lowest PSNR, since LA turns the noises in Hall-monitor into large-area flickering. 3FMA exhibits higher PSNR than HMDEPR due to some false motion introduced by the dilation filter. HMDEPR provides the best result in foreman, as it performs correct motion detection of fast motion, sharp edge interpolation, and good interpolation of semimoving bricks and trees.

The intra-interpolation is very important in the sequences with global motion such as bus, coastguard, mobile, Stefan, and vectra-color. The horizontal motion in coastguard favors dedicated 4FHMD. LA benefits the interpolation of fast moving textures, which appear in bus and Stefan, All motion-adaptive algorithms, even 2FMA, suffer from motion missing problem in these two sequences. Our HMDEPR retains a better result than the other motion-adaptive algorithms because of the smaller motion missing ratio and the aids of EPR for texture interpolation. Vectra-color contains not only fast global motion but also many sharp edges, which makes HMDEPR better than LA. The slow motion of mobile benefits 3FMA, enabling the most accurate motion detection without false motion. Although LA sometimes outperforms HMDEPR in the fast moving scene, HMDEPR still possess the capability in interpolating edges as discussed in Section 3.3 and indicated in Figure 11. The average PSNR of HMDEPR is the highest among all algorithms, which explains the robustness and content-adaptability of HMDEPR.

4. CONCLUSION

This paper presents a novel motion-adaptive deinterlacer which incorporates new hybrid motion detection and the edge-pattern recognition for intra-interpolation. The hybrid motion detector, which combines the benefits of 2-field and 3-field motion detection, is capable of detecting slow motion, fast motion, and the motion of edges with high accuracy. The edge-pattern recognition algorithm performs local scene analysis and adaptive interpolation, and thus achieves successful interpolation of textures and edges which cannot be accomplished by using LA or ELA along. The edge-pattern recognition also introduces the feasibility of using motion-adaptive method in not only pixel interpolation but also pixel prediction for scene analysis.

We compare our deinterlacing algorithm to six algorithms, including two recently published algorithms with 4-field motion detection. Versatile video contents, which include stationary textures, moving textures, fast motion, and edges are adequately processed by our algorithm as indicated in the comparison of visual quality. The key concept of motion-adaptive deinterlacing is to adaptively accommodate stationary and moving contents. The PSNR of our deinterlacer on versatile sequences demonstrates higher robustness than the other motion-adaptive algorithms. Moreover, with better performance than the 4-field motion-adaptive algorithms, our algorithm only needs the data of three fields, which reduces the memory cost in VLSI implementation. The cost and performance comparison justifies the efficiency and content-adaptability of our algorithm.

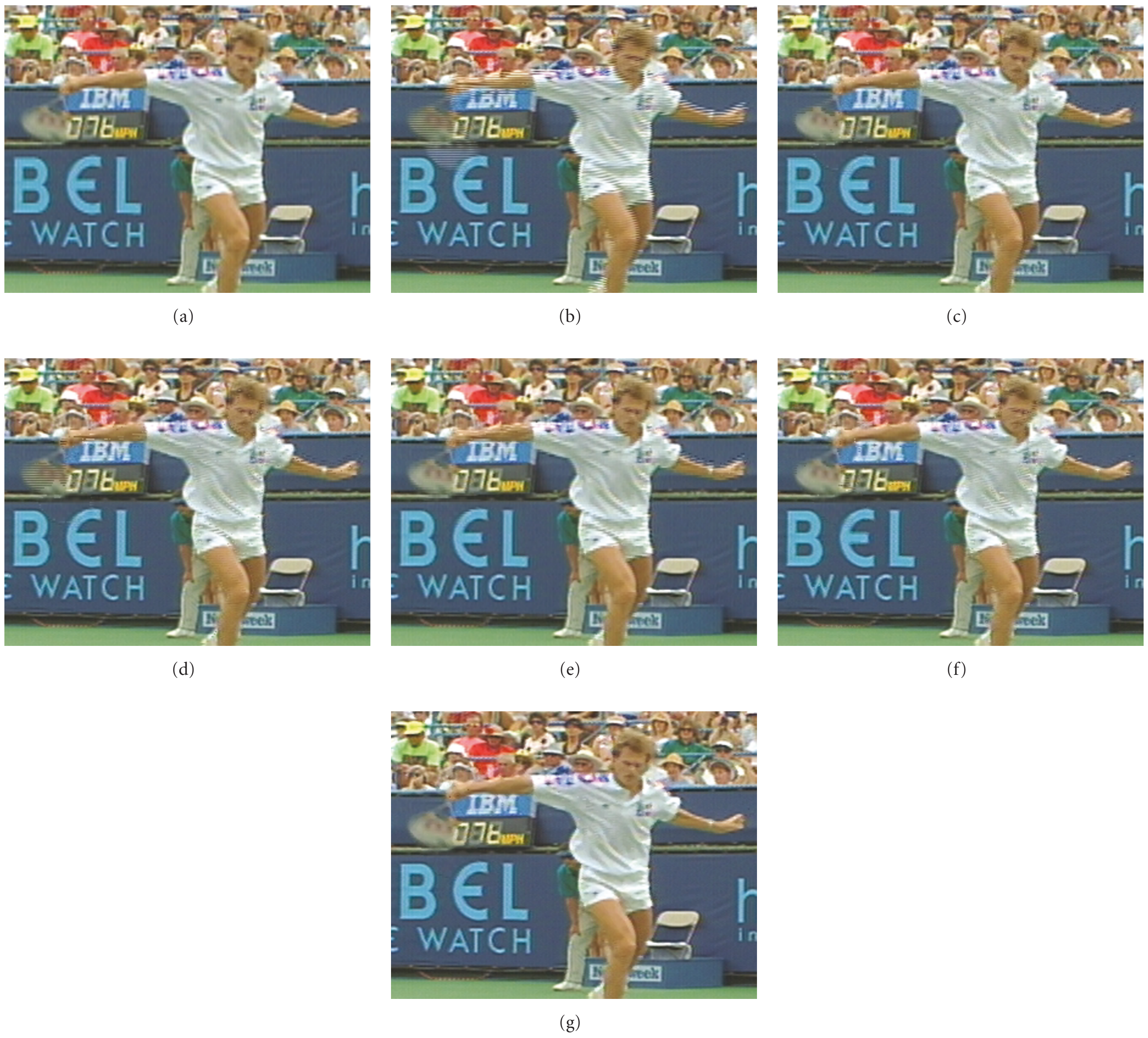


Figure 10: Visual quality of 50th frame of Stefan. (a) LA, (b) FI, (c) 2FMA, (d) 3FMA, (e) 4FTD, (f) 4FHMD, (g) proposed HMDEPR.

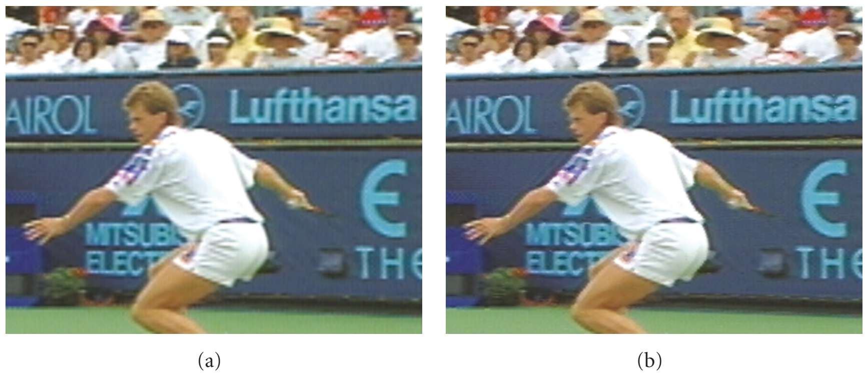


Figure 11: Visual quality of 194th frame of Stefan. (a) LA, (b) proposed HMDEPR.

Table 4: PSNR of the deinterlacing algorithms in dB.

	Total picture number	LA	FI	2FMA	3FMA	4FTD [21]	4FHMD [5]	HMDEPR
Bus	150	28.19	20.60	26.83	25.37	26.86	25.24	26.87
Coastguard	300	28.67	27.27	28.43	29.91	29.63	30.41	29.13
Foreman	300	32.39	28.74	31.34	32.42	32.50	32.84	33.93
Hall monitor	300	31.76	36.70	32.49	39.58	35.07	36.69	38.79
Mobile	300	25.47	23.49	25.21	26.70	25.99	24.55	24.67
M&D	300	39.20	40.54	39.02	42.59	41.90	40.91	43.23
Silent	300	33.80	34.42	35.93	39.31	37.95	38.68	41.10
Stefan	300	27.46	21.01	26.59	25.82	25.83	25.56	26.88
Vectra-color	142	26.63	22.28	26.42	25.69	27.05	26.75	27.31
Average		30.40	28.34	30.25	31.93	31.42	31.29	32.43

ACKNOWLEDGMENT

This work was supported by National Science Council, Taiwan, under Contract no. NSC95-2221-E006-481: "Advanced Electronic System Level Research and Design for High Definition Video Frame Rate Conversion" in 2006.

REFERENCES

[1] E. Dubois, G. de Haan, and T. Kurita, "Motion estimation and compensation technologies for standards conversion," *Signal Processing: Image Communication*, vol. 6, no. 3, pp. 189–190, 1994.

[2] G. de Haan and E. B. Bellers, "Deinterlacing—an overview," *Proceedings of the IEEE*, vol. 86, no. 9, pp. 1839–1857, 1998.

[3] T. Doyle and M. Looymans, "Progressive scan conversion using edge information," in *Signal Processing of HDTV II*, L. Chiariglione, Ed., pp. 711–721, Elsevier, Amsterdam, The Netherlands, 1990.

[4] C. J. Kuo, C. Liao, and C. C. Lin, "Adaptive interpolation technique for scanning rate conversion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 317–321, 1996.

[5] S.-F. Lin, Y.-L. Chang, and L.-G. Chen, "Motion adaptive interpolation with horizontal motion detection for deinterlacing," *IEEE Transactions on Consumer Electronics*, vol. 49, no. 4, pp. 1256–1265, 2003.

[6] Y.-L. Chang, S.-F. Lin, and L.-G. Chen, "Extended intelligent edge-based line average with its implementation and test method," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '04)*, vol. 2, pp. 341–344, Vancouver, BC, Canada, May 2004.

[7] B. Bhatt, F. Templin, A. Cavallerano, et al., "Grand alliance HDTV multi-format scan converter," *IEEE Transactions on Consumer Electronics*, vol. 41, no. 4, pp. 1020–1031, 1995.

[8] A. M. Bock, "Motion-adaptive standards conversion between formats of similar field rates," *Signal Processing: Image Communication*, vol. 6, no. 3, pp. 275–280, 1994.

[9] D. Han, C.-Y. Shin, S.-J. Choi, and J.-S. Park, "A motion adaptive 3-D de-interlacing algorithm based on the brightness profile pattern difference," *IEEE Transactions on Consumer Electronics*, vol. 45, no. 3, pp. 690–697, 1999.

[10] T. Koivunen, "Motion detection of an interlaced video signal," *IEEE Transactions on Consumer Electronics*, vol. 40, no. 3, pp. 753–760, 1994.

[11] G. G. Lee, D. W.-C. Su, H.-Y. Lin, and M.-J. Wang, "Multiresolution-based texture adaptive motion detection for deinterlacing," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '06)*, pp. 4317–4320, Island of Kos, Greece, May 2006.

[12] D. Hargreaves and J. Vaisey, "Bayesian motion estimation and interpolation in interlaced video sequences," *IEEE Transactions on Image Processing*, vol. 6, no. 5, pp. 764–769, 1997.

[13] K. Sugiyama and H. Nakamura, "A method of de-interlacing with motion compensated interpolation," *IEEE Transactions on Consumer Electronics*, vol. 45, no. 3, pp. 611–616, 1999.

[14] M. Biswas and T. Nguyen, "A novel de-interlacing technique based on phase plane correlation motion estimation," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '03)*, vol. 2, pp. 604–607, Bangkok, Thailand, May 2003.

[15] Y.-Y. Jung, S. Yang, and P. Yu, "An effective de-Interlacing technique using two types of motion information," *IEEE Transactions on Consumer Electronics*, vol. 49, no. 3, pp. 493–498, 2003.

[16] G. G. Lee, K. A. Vissers, and B.-D. Liu, "On a 3D recursive motion estimation algorithm and architecture for digital video SoC," in *Proceedings of the 47th Midwest Symposium on Circuits and Systems (MWSCAS '04)*, vol. 2, pp. 449–451, Hiroshima, Japan, July 2004.

[17] G. G. Lee, M.-J. Wang, H.-Y. Lin, D. W.-C. Su, and B.-Y. Lin, "A 3D spatio-temporal motion estimation algorithm for video coding," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '06)*, pp. 741–744, Toronto, Ontario, Canada, July 2006.

[18] G. G. Lee, M.-J. Wang, H.-Y. Lin, D. W.-C. Su, and B.-Y. Lin, "Algorithm/architecture co-design of 3-D spatio-temporal motion estimation for video coding," *IEEE Transactions on Multimedia*, vol. 9, no. 3, pp. 455–465, 2007.

[19] J. E. James Jr., "Interactions between color plane interpolation and other image processing functions in electronic photography," in *Cameras and Systems for Electronic Photography and Scientific Imaging*, vol. 2416 of *Proceedings of SPIE*, pp. 144–151, San Jose, Calif, USA, February 1995.

[20] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, Prentice-Hall, Upper Saddle River, NJ, USA, 2nd edition, 1999.

[21] Y. Shen, D. Zhang, Y. Zhang, and J. Li, "Motion adaptive deinterlacing of video data with texture detection," *IEEE Transactions on Consumer Electronics*, vol. 52, no. 4, pp. 1403–1408, 2006.