A MOTION ADAPTIVE DEINTERLACING ALGORITHM USING IMPROVED MOTION DETECTION

Janghyeok Yun, Jeehyun Jun, Gwanggil Jeon, and Jechang Jeong

1Department of Electronics and Computer Engineering, Hanyang University
22 Wangsimni-ro, Seongdong-gu, Seoul 133-791, Korea
2Department of Embedded Systems Engineering, University of Incheon
12-1 Songdo-dong, Yeonsu-gu, Incheon 406-772, Korea
yjh2748@hanyang.ac.kr, happygp2@gmail.com, gjeon@incheon.ac.kr, jjeong@hanyang.ac.kr

ABSTRACT
In this paper, a motion adaptive deinterlacing algorithm is proposed. It consists of three parts: (1) modified edge-based line average, (2) pixel-based consequent five-field motion detection, and (3) block-based local characteristic for detecting true motion and calculating the motion intensity by using an improved method which is able to detect the inner part of moving objects precisely as well as to reduce the risk of false detection caused by intrinsic noises in the image. Depending on the detected motion activity level, it combines spatial and temporal methods with weighting factor. Simulations conducted on several video sequences indicate that the performance of the proposed method is superior to the conventional methods in terms of both subjective and objective video quality.

KEY WORDS
Video deinterlacing, Interlaced-to-progressive format conversion, Motion adaptive, Motion detection

1. Introduction
In video broadcasting, general TV systems such as PAL, SECAM, and NTSC currently adopt an interlaced format to halve the video transfer bandwidth. This standard frame rate is sufficient for distributing slow motion; however, it tends to introduce flickering artifacts for objects that have high horizontal frequencies which includes interline flicker, jaggedness, and line crawling [1,2]. In addition, the growth of modern display systems such as high definition television (HDTV), PC monitor, liquid crystal displays (LCD), and plasma display panels (PDP) require that the whole image be displayed at once.

Deinterlacing is a converting technique that changes the interlaced scan format to the progressive scan format. Deinterlacing methods are generally classified into four classes: intra-field, inter-field, motion adaptive and motion compensated methods. Intra-field methods [3-6] use the pixels of the current field to fill in missing lines. These methods work well in video sequences with low spatial frequency, however they produce annoying flickering artifacts in areas with high spatial frequencies. Inter-field methods [7,8] exploit high correlation between adjacent fields in order to improve video quality. They provide the best quality in still areas but can produce combing artifacts in moving areas. Motion adaptive (MA) methods [9-12] combine the advantages of intra-field and inter-field method. MA methods are among the most popular deinterlacing method owing to the low complexity and high video quality. Motion compensated (MC) methods [13,14] are has to calculate the displacement of blocks or objects within neighboring fields. MC methods achieve better performance than other methods, but they require high computational cost.

This paper is organized as follows. In Section 2, an overview of motion adaptive deinterlacing method is presented. Section 3 presents the proposed deinterlacing algorithm. Section 4 shows experimental results and the comparison with previous methods. Section 5 concludes the paper.

2. Overview of Motion Adaptive Deinterlacing Method
The MA deinterlacing combines the advantages of both intra-field and inter-field deinterlacing methods. It detects the motion areas first. Then, it adopts intra-field deinterlacing method in the motion area and Weave deinterlacing method in the static areas. Motion possibility values, calculated after motion detection step, are used as the weighting factor for combining the results of spatial and temporal interpolations to improve the performance. The block diagram of MA deinterlacing is
shown in Fig. 1. The interpolation step could be formulized by Eq. (1).

$$D_n(i,j) = \begin{cases} 
X_n(i,j), & \text{if } (y \mod 2 = n \mod 2), \\
(P_M \cdot X_{\text{max}}(i,j) + (1 - P_M) \cdot X_{\text{max}}(i,j)), & \text{otherwise}
\end{cases}$$

where, $P_M$ is the motion possibility value, $(i,j)$ represents the pixel position, $n$ represents current field, $X_n(i,j)$ represents the pixel value at position $(i,j)$, and $D_n(i,j)$ represents the final interpolated pixel at the same position. Therefore we can obtain a progressive video.

### 3. Proposed Algorithm

In this section, we introduce a deinterlacing algorithm which contains pixel- and block-based motion detection algorithm and decision weighting factor ($W$). And then, spatial interpolation, temporal interpolation and combined interpolation methods with weighting factor and motion intensity are presented.

#### 3.1 Motion Detection

##### 3.1.1 Five-field pixel-based motion detection

![Figure 2. Conventional motion detection algorithm](image)

As mentioned in the previous section, motion adaptive techniques are highly dependent on the accuracy. Typically, there are high correlations between the current frame and adjacent frames. We can obtain motion information by comparing two or more reference fields. Figure 2 shows conventional five-field pixel-based motion detection algorithm. It supposes $n$ is current field and $X$ is the pixel to be interpolated. Note that the pairs $(n-2,n)$, $(n-1,n+1)$ and $(n,n+2)$ have same missing lines and the same time difference.

In this algorithm, the motion intensity $MI_1$, $MI_2$, $MI_3$ are calculated respectively as follows:

$$MI_1 = |C - F|,$$
$$MI_2 = \left| \frac{(A + B) - (D + E)}{2} \right|,$$
$$MI_3 = \left| \frac{(D + E) - (G + H)}{2} \right|.$$ (2)

$MI$ is defined as Eq. (3).

$$MI = \max(MI_1, MI_2, MI_3).$$ (3)

However this algorithm cannot detect the moving objects precisely. It produces noise and can wrongly detect the inside of moving objects. To increase the ability of detecting motions, we exploit FIR filter as shown Fig. 3. It includes the neighboring right and left side of each of two pixels in motion detection. Comparing conventional algorithm, its pixel values are calculated as shown in Eq. (4).

$$A = \frac{(A_1 + 2A_2 + 4A_3 + 2A_4 + A_5)}{10},$$ $$B = \frac{(B_1 + 2B_2 + 4B_3 + 2B_4 + B_5)}{10},$$ $$C = \frac{(C_1 + 2C_2 + 4C_3 + 2C_4 + C_5)}{10},$$ $$D = \frac{(D_1 + 2D_2 + 4D_3 + 2D_4 + D_5)}{10},$$ $$E = \frac{(E_1 + 2E_2 + 4E_3 + 2E_4 + E_5)}{10},$$ $$F = \frac{(F_1 + 2F_2 + 4F_3 + 2F_4 + F_5)}{10},$$ $$G = \frac{(G_1 + 2G_2 + 4G_3 + 2G_4 + G_5)}{10},$$ $$H = \frac{(H_1 + 2H_2 + 4H_3 + 2H_4 + H_5)}{10}.$$ (4)

$MI_1, MI_2, MI_3, MI_4, MI_5$ are:

$$MI_i = |A - D|,$$
$$MI_i = |B - E|,$$
$$MI_i = |C - F|,$$
$$MI_i = |D - G|,$$
$$MI_i = |E - H|. ($5$)

$MI$ is defined as Eq. (6).

$$MI = \max(MI_1, MI_2, MI_3, MI_4, MI_5).$$ (6)
3.1.2 Three-field block-based motion detection

By comparing the pixel value between fields, we can determine the movement’s type of object in each field. However, this method fails to detect low movement of an object. For more accurate motion estimation in low-movement region, we add an additional block-based method. Figure 4 shows the process of calculating the sum of absolute difference (SAD) of 3×3 blocks.

In order to evaluate SAD values, empty pixels (yellow circles) have to be determined in order to execute the 3×3 block. Line average method is used in this preprocessing step. Three pairs of SAD are calculated at \((n-1,n), (n,n+1), (n-1,n+1)\) fields, respectively. Formula is defined as follows:

\[
\begin{align*}
\text{SAD}_{\text{cur-prev}} &= \sum_{i=-1}^{1} \sum_{j=-1}^{1} |X_{n}(i,j) - X_{n-1}(i,j)|, \\
\text{SAD}_{\text{cur-next}} &= \sum_{i=-1}^{1} \sum_{j=-1}^{1} |X_{n}(i,j) - X_{n+1}(i,j)|, \\
\text{SAD}_{\text{next-prev}} &= \sum_{i=-1}^{1} \sum_{j=-1}^{1} |X_{n+1}(i,j) - X_{n-1}(i,j)|.
\end{align*}
\] (7)

3.2 Weighting Factor Calculation

Using the \(MI\) and three SAD values obtained from the motion detection stage, we calculate interpolation coefficient \(W\) and \((1-W)\). \(W\) is used as ratio factor when combining results of spatial and temporal interpolation. Calculation weighting factor consists of two steps.

**Step 1:**
Weighting factor is calculated as shown Eq. (8).

\[
W = \begin{cases} 
0, & \text{if } (MI \leq \tau_1), \\
1, & \text{if } (MI \geq \tau_2), \\
\frac{MI - \tau_1}{\tau_2 - \tau_1}, & \text{otherwise} 
\end{cases}
\] (8)

where \(\tau_1\) and \(\tau_2\) are decided empirically. If the motion intensity is smaller than threshold 1(\(\tau_1\)), this pixel is considered as static part of video or stationary scene and, therefore, inter-field interpolation (temporal interpolation) is selected. If the motion intensity is greater than threshold 2(\(\tau_2\)), intra-field interpolation is adopted. If \(MI\) is between \(\tau_1\) and \(\tau_2\), a mixture of spatial and temporal interpolation is selected. Figure 5 shows the graph of weighting factor.
Step 2:
We add an extra condition to refine the weighting factor. If \( W \) obtained from Step 1 is zero, \( W \) with zero value is considered having no movement pixel at Step 1. However, among them, there could be some pixels which are considered as stationary area due to low MI they might actually have slight movements. If three SAD values are greater than the threshold 3 (\( \tau_3 \)), \( W \) is set to 1 because we assume there are subtle movement, and spatial filtering is performed.

\[
\begin{align*}
\text{If} & \quad \left( W = 0 \right) \land \left( \text{SAD}_{\text{cur-prev}} > \tau_3 \right) \land \left( \text{SAD}_{\text{next-prev}} > \tau_3 \right), \\
\text{Then, we updated } W & \text{ as } W = 1. \\
\text{Otherwise}, & \quad W \text{ is not changed.}
\end{align*}
\]

3.3 Spatial Interpolation (Intra-Field Interpolation)
In the spatial interpolation, we use modified edge-based line average (MELA) algorithm [6]. The MELA method is a modified version of efficient edge-based line average (EELA) [5]. Instead of using absolute difference between two pixels in three directions to detect edge direction, MELA additionally considers three direction vectors, which cover vertical (90°), diagonal (63°), and anti-diagonal (117°) directions, as follows:

\[
\begin{align*}
P & = \frac{\left| U_{i,j} - L_0 \right| + \left| U_0 - L_{i,j} \right|}{2}, \\
Q & = \frac{\left| U_0 - L_{i,j} \right| + \left| U_{i,j} - L_{0} \right|}{2}, \\
V & = \frac{\left| U_{i,j} - L_0 \right| + \left| U_{i,j} - L_{0} \right| + \left| U_{i,j} - L_{i,j} \right|}{4}.
\end{align*}
\]

(9)

The pixel correlations of the edge-based line average (ELA) [4] are then calculated as follow:

\[
\begin{align*}
C_1 & = \left( U_{i,j} - L_{i,j} \right), \\
C_0 & = \left( U_0 - L_0 \right), \\
C_1 & = \left( U_{i,j} - L_{i,j} \right).
\end{align*}
\]

(10)

4. Experimental Results
To evaluate performance of the proposed deinterlacing method, we conducted experiments with various standard CIF video sequences including from low motion to dynamic videos. The threshold values used in the simulations are set as \( \tau_1=20, \tau_2=70, \tau_3=100 \). In Figs. 7(a) and 7(b) are the results of applying Step 1 and Step 2, respectively. In Fig. 7(a), the ball is moving downwards.
The ball is existing in \((n-1)\) and \((n+1)\) fields, but is absent in \((n-2)\), \((n)\), and \((n+2)\) fields. In case there is only
one moving object, the pixel-based motion detection can often wrongly detect the motion. Figure 8 shows similar, and one of the worst cases. The letters are moving from right to left, but, almost of the pixels are considered as
Therefore temporal filter is selected, although there must be many movements. To improve this situation, we apply the Step 2 and can show result of motion detection well, in Figs. 7 (b) and 8 (b). Table 1 shows the PSNR comparison and the subjective quality comparison are shown in Figs. 9, 10 and 11. The proposed deinterlacing method has better performance than several well-known deinterlacing algorithms such as LA, ELA, MELA, vertical temporal filter methods and motion adaptive methods for deinterlacing. Proposed method achieved around 0.95dB increase in PSNR. Particularly, objective performance shows much better result in low motion videos such as Akiyo, Mother&daughter, and Container than high motion videos. In Figs. 9, 10, and 11, fixed pixels are marked with white rectangles. These parts are relatively low motion areas. The other methods are blurred a little bit, while the proposed method has clear interpolation results.

5. Conclusion

We present a motion adaptive deinterlacing algorithm with new motion detection method. In order to detect motion precisely, five-field pixel based motion detection using FIR filter and three-field block based motion detection methods are introduced. After we have calculated the weighting factor and figured out the conditions of the pixel to be interpolated, it is possible to apply temporal and spatial filters, which gives accurate interpolated pixel value. Experimental results show that the proposed deinterlacing method outperformed the other methods in terms of subjective and objective evaluations.

Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2012-0001584)

References


