Locally Estimated Heterogeneity Property and Its Fuzzy Filter Application for Deinterlacing

Gwanggil Jeon[1], Marco Anisetti[2], Lei Wang[1], and Ernesto Damiani[3]
[1]Department of Embedded Systems Engineering, Incheon National University, Incheon, Korea
[2]Dipartimento di Informatica, Università degli Studi di Milano, Via Bramante 65, 26013 Crema (CR), Italy
[3]Department of Electrical and Computer Engineering, Khalifa University, P.O.Box 127788, Abu Dhabi, UAE
email: gjeon@inu.ac.kr, marco.anisetti@unimi.it, lwang@inu.ac.kr, ernesto.damiani@kustar.ac.ae
Tel: +82-32-835-8946, Fax: +82-32-835-0782

Abstract

This paper presents an intra-field scanning format conversion method using two filters: bilinear filter (BF) and fuzzy-based weighted average filter (FWAF). The proposed method is intended for black and white images, luminance component of YIQ color space, or each color component of RGB color space. We start from the notion that pixels to be interpolated can be classified into two areas based on local variance: homogeneous and heterogeneous areas. According to the local variance criteria, we apply the FWAF to the heterogeneous area and the BF to the homogeneous one, producing good visual results. Our FWAF consists of an intensity similarity filter and a geometric closeness filter. The latter is used to populate the heterogeneous area with the missing lines, due to its high deinterlacing precision. Our experimental results show that the proposed approach provides satisfactory performances in terms of both objective metrics and visual image quality. We used parameter tuning on our training set to explore the relationship between objective quality and computational complexity. We report on how to achieve good performance or the best quality-speed tradeoff using the methods researched.

Keywords: Deinterlacing, fuzzy-based weighted average filter, variance

estimation, window characteristic, intensity similarity, geometric closeness.

1. Introduction

The fuzzy concept-based methods can model uncertainty and subjective concepts in image processing [1]. Edge details are key factors for improving the subjective perception of an image. However, concluding whether a pixel belongs to a homogeneous or heterogeneous areas is not a trivial work. This study focuses on the development of fuzzy metric-based methods for the particular task of video deinterlacing. Present digital television transmission formats use an interlaced scan mode. The high-definition television (HDTV) broadcasting system, such as ATSC and DVB, accepts an interlaced scanning format ($1080i$, 1080×1920 resolution with only 540 lines scanned in each frame) [2], where 'i' stands for interlaced scanning. Interlaced scanning is directly compatible with some CRT-based HDTV sets where video can be displayed natively in interlaced form, but for display on modern progressive-scan LCD and PDP sets, video must be deinterlaced and in many cases, scaled to the display resolution.

Interlaced scan fields contain half the samples of the original signal, and only the even or odd lines of a frame are scanned and displayed serially. The idea of interlaced scan was considered in the first place because of a well-known fact of human physiology: the human visual system is more sensitive to flicker, serration and line crawl when screens get bigger and brighter, and the frame rate become higher [3; 4]. As displays become larger and brighter, there is a necessity for conversion between interlaced and progressive scanning formats. The purpose of interlaced scanning is to accomplish a tradeoff between the frame rate and transmission bandwidth requirements [5]. The conversion process from interlaced fields into progressive frames is called *deinterlacing*.

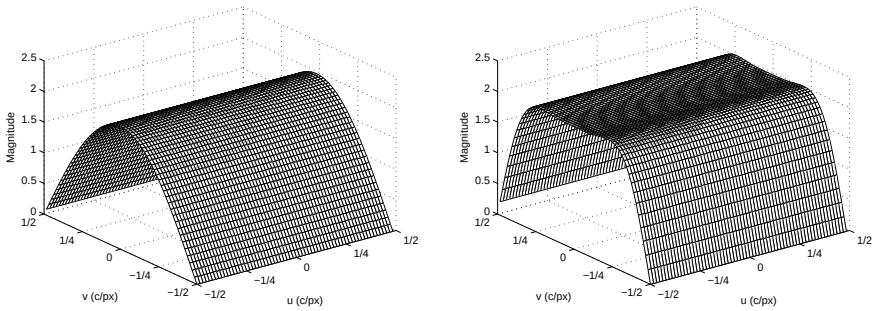
Conventional deinterlacing populates the missing lines in two ways: intra-field methods and inter-field methods. Inter-field methods can be further categorized into non-motion compensated (NMC) and motion compensated (MC) methods [6-11]. Inter-field methods use not only current field but also neighbor frames, and as a result they provide better image quality with less motion scenes. However, the NMC methods are not able to correctly deinterlace sequences with high spatial motion frequencies. As the human visual system is very sensitive to details, even a single badly interpolated

edge may considerably lower the visual quality of the results. In turn, MC methods provide good results in general. However, due to processing motion information, higher complexity than the intra-field methods and NMC. Intra-field methods need less computational resources than motion-based inter-field methods because they only use the current field and are therefore, more reliable for real-time applications.

Intra-field methods can be classified into two categories: edge direction-based and filter-based methods. The former compute dominating edge direction along which to deinterlace with a skewed line average filter. Some examples of edge direction-based methods are edge map-based deinterlacing (EMD) [12], low-complexity interpolation method for deinterlacing (LCID) [13], modified ELA (MELA) [14], deinterlacing using locally adaptive-threshold binary image (LABI) [15], and fine edge-preserving deinterlacing (FEPD) [16]. However, all edge direction-based approaches suffer from occasional low performance as a result of incorrect directional estimation or the limitations of direction models in high spatial frequency areas or horizontal edges. EMD, LCID, MELA, and FEPD sometimes yield incorrect edge direction because they consider only horizontal and vertical gradients to compute the local edge direction. The LABI method provides noticeable improvements on the specific regions where horizontal edges exist, but is computationally heavy due to its large search range.

The other category is filter-based methods. Some examples of this category are modified covariance-based adaptive deinterlacing (MCAD) [17], local surface model-based deinterlacing (LSMD), and least squares method based frequency domain filter-based deinterlacing (FFD). As long as the similarity between the high-resolution covariance and the low-resolution covariance is firmly settled, the optimal linear interpolation coefficients for minimum mean squared error (MSE) can be extracted by the classical Wiener filter (MCAD and LSMD) and least squares filter (FFD). However, the major drawback of MCAD and LSMD is their pricey computational complexity. To alleviate this issue, FFD computes filter coefficients before the implementation of deinterlacing by pre-processed training. However, the obtained filters do not always guarantee the minimum MSE results.

In this paper, we present our novel intra-field deinterlacing method. We first classify pixels into two regions, homogeneous and heterogeneous, using local variance criteria. The area with higher local variance is called *heterogeneous* and a novel fuzzy-based weighted average filter (FWAF) is applied to it. In turn, the area with lower local variance is designated as *homogeneous*



(a) (b)

Figure 1: Perspective view of frequency responses of filters for deinterlacing: (a) h_{2TF} and (b) h_{6TF}.

and a bilinear filter (BF) is applied to it. FWAF consists of intensity similarity filter and geometric closeness filter, which is employed to interpolate the missing pixels. Finally, all weights assigned to neighbor pixels are considered for populating the missing pixels.

This paper is organized as follows. Our proposed method is described in Section 2, where the fuzzy filter approach, the variance estimation method for local windows, and the implementation of deinterlacing are explained. In Section 3, experimental results and performance analyses are discussed to show the reliability of the proposed method. Finally, Section 4 draws our conclusions.

2. Proposed Algorithm

2.1. Filter-based Approach

In this paper, we focus on an intra-field method which belongs to the NMC category and provides good performance with low complexity. The bilinear interpolation (Bob) method uses a single field to restore a progressive frame along orthogonal (90^o) edge directions. The edge-based method uses the uniform weighted sum of 2-Tap Filter (2TF), $h_{2TF} = [1\ 1]/2$, to reconstruct the missing pixel along a determined edge direction. The frequency response of 2TF looks like a bell shape as shown in Fig. 1(a). For this reason, high frequency information is not well restored, and 2TF may cause apparent jaggedness at the edges of the area.

The *sinc* function defines an ideal filter whose frequency response is a rectangular shape, with vertical frequency cuts. Based on this function, we can design a real filter having a steeper frequency cuts than the 2TF, so interlaced signals can be reconstructed more accurately. In [18], the authors adopted *sinc* filter, which is a 1D 6-Tap Filter ($h_{6TF} = [3\ -17\ 78\ 78\ -17\ 3]/128$) as shown in Fig. 1(b). Coefficients of 'h_{6TF}' are determined by approximating the *sinc* function.

We remark that this is the same method used in HEVC to decrease residual errors. However, this method only deals with similarity of *sinc* function, and topological parameters like closeness or spatial locality are not taken into account.

Let (i, j) be the spatial Cartesian coordinates of each pixel in an original interlace image X_{in} of size $S_V \times S_H$, $x_{(i,j)}$ be the gray level intensity of the pixel in the position (i, j), where $0 \leq x_{(i,j)} \leq max - 1$ and max is the maximum number of gray level intensity in the image ($max = 2^8$). As X_{in} is an interlace image, the vertical resolution of X_{in} is halved and we assume vertically even numbered pixels in X_{in} are missing. The intensity of $x_{(i,j)}$ ranges $[0, max - 1]$, or alternatively, it is standardized to the interval $[0, 1]$ in order to fuzzify the image utilizing the fuzzy inference system. The pixels which are located in a window W of size $M \times N$ centered in (i, j) are described by, $x^k_{(i+m,j+n)}$ for $k = 0, \ldots, M \times N - 1$, where vertical and horizontal sizes M and N are odd numbers ($M, N \geq 3$), parameters m and n are vertical and horizontal pixel displacements, respectively, which meet $-\lfloor M/2 \rfloor \leq m \leq \lfloor M/2 \rfloor$ and $-\lfloor N/2 \rfloor \leq n \leq \lfloor N/2 \rfloor$. The center pixel $x_{(i,j)}$ is simply remarked p^0. Then the pixel set of adjacent pixels of $x_{(i,j)}$ within a window W is noted $p^1, p^2, \ldots, p^{M \times N - 1}$. For example, when $(M, N) = (3, 3)$, all pixels in a window W are labeled as shown in Fig. 2 and Eq. (1).

$$W = \begin{bmatrix} p^1 & p^2 & p^3 \\ p^8 & p^0 & p^4 \\ p^7 & p^6 & p^5 \end{bmatrix} \tag{1}$$

In recent years, fuzzy logic-based filters have shown to be able to support effective image filtering [11; 19; 20]. Fuzziness is introduced as a fuzzification of classical filters, and achieves a fuzzy weighted combination of the outputs of several subfilters. We introduce fuzzification via rules of inference which are designed to be directly applied to the pixels, $x_{(i,j)}$. In this paper, the FWAF filter is employed in order to restore the missing pixels in the interlaced

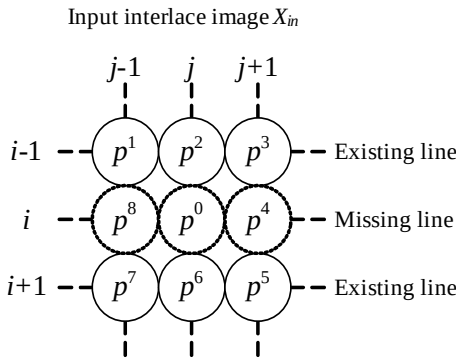


Figure 2: Existing and missing pixels in a 3×3 window W of input interlace image X_{in}. Circle with dotted line are the missing pixels.

images. Our FWAF is based on weighted average function between a center pixel and its adjacent pixel. As described below, our fuzzy rules provide the inputs to the fuzzy inferential filters composing the FWAF.

These filters minimize the error between original and the reconstructed images in terms of a distance. Their output undergoes a defuzzification process, which merges the effects of the used rules. Finally, a hybrid filter approach is used which combines FWAF and a conventional BF. The FWAF is used in heterogeneity window (W_{HE}), and BF filter is used in homogeneity window (W_{HO}). Before starting the process, we initialize the missing pixels in the window W, i^{th} row ($p^8 = x_{(i,j-1)}$, $p^0 = x_{(i,j)}$, $p^4 = x_{(i,j+1))}$ as follows:

$$p^8 = \frac{p^1 + p^7}{2}, \quad p^0 = \frac{p^2 + p^6}{2}, \quad p^4 = \frac{p^3 + p^5}{2}. \tag{2}$$

2.2. Variance Estimation for Local Window

The given image is firstly separated into $N \times N$ size window W, centered in (i, j). Let us consider $N = 3$, we will discuss how N is obtained empirically in Section 3.2. We apply mean and variance calculation equations to obtain μ and σ^2 for window W. Following local image analyzer is used to determine the degree of variance for a window W. All pixels in a window $W = \{p^k\}_{k\in\{0,\ldots,8\}}$ are considered to follow independent and identically distributed characteristics. With the independent and identically-distributed characteristic, the mean value of W, μ_W, is empirically obtained as,

$$\mu_W = \frac{1}{9}\sum_{k=0}^{8} p^k. \tag{3}$$

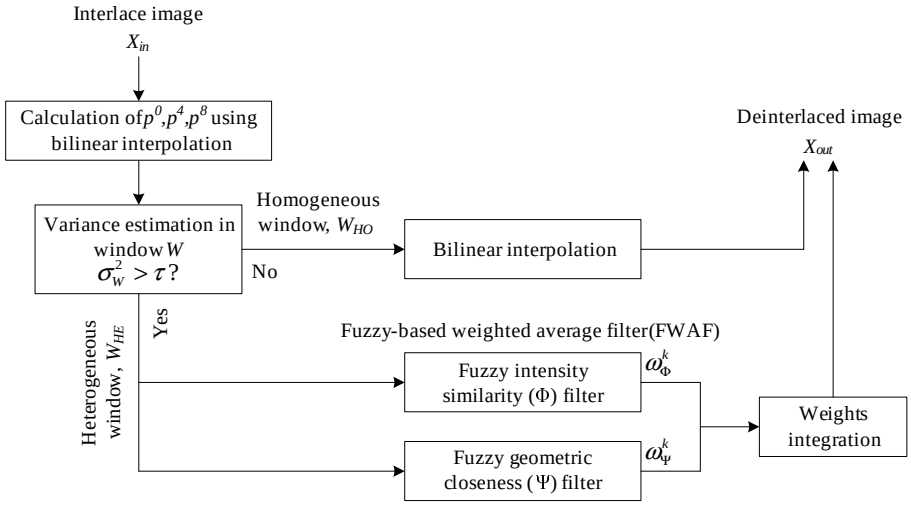


Figure 3: Block diagram of the proposed method.

As μ_W stands for the mean of window W, we assume the variance of W, σ_W^2, is considered to symbolize the local variance of the window W. We assume the σ_W^2 measure is approximately constant, which is computed as follows,

$$\sigma_W^2 = \frac{1}{9}\sum_{k=0}^{8}\left(p^k - \mu_W\right)^2. \tag{4}$$

With the information of window variance σ_W^2 and threshold value τ, a given W is determined as W_{HE} or W_{HO}. When σ_W^2 is smaller than threshold τ, we assume local window W is homogeneity ($W = W_{HO}$). On the other hand, when σ_W^2 is bigger than threshold τ, local window W is considered as heterogeneity ($W = W_{HE}$).

$$\begin{array}{ll} W = W_{HO} & \text{if } \sigma_W^2 \leq \tau \\ W = W_{HE} & \text{else} \end{array} \tag{5}$$

The threshold parameter τ is obtained empirically, as discussed in Section 3.3.

2.3. Deinterlacing Implementation

As described in preceding subsection, our FWAF is only used in W_{HE}. To generate the FWAF, we need two sub-filters: intensity similarity (Φ)

filter and geometric closeness (Ψ) filter. Both filters are dependent on fuzzy operators, which evaluate and return weights to pixels to reconstruct the original image structure. Each filter is applied to the interlace image, X_{in}, separately. The fuzzy operator is designed to restore full resolution image, X_{out}, in accordance with the given window W.

One of the key factors of our method is to combine gray levels using intensity similarity filter and geometric closeness filter. Both weight-functions regarding the pixel intensity distance and geometric distance are intended to replace the center pixel value, p^C, with the average of the similar and adjacent intensity values in the given window. Using the pixel information located in the window W_{HE} centered at (i, j), the weighted average filter is applied to calculate output value p^C by a linear combination of all pixels p^k,

$$p^C = \sum_{k=1}^{8} \frac{\omega_\Phi^k \omega_\Psi^k p^k}{\omega_\Phi^k \omega_\Psi^k}, \tag{6}$$

where Φ and Ψ are filter categories, ω_Φ^k and ω_Ψ^k are weights of each category, and p^k is k^{th} pixel in window W_{HE}. Both fuzzy weights ω_Φ^k and ω_Ψ^k are found by employing a fuzzy membership function specified by a distance criterion, $d_\Phi(\cdot)$ and $d_\Psi(\cdot)$. A sigmoid function (SF) is used as distance criterion which is adopted to determine the weights,

$$\begin{aligned} w_\Phi^k &= SF(d_\Phi(p^k)) = \frac{1}{1 + e^{d_\Phi(p^k)}}, \\ w_\Psi^k &= SF(d_\Psi(p^k)) = \frac{1}{1 + e^{d_\Psi(p^k)}}, \end{aligned} \tag{7}$$

where $k = 1, 2, \ldots, 8$, weighting functions $d_\Phi(p^k)$ and $d_\Psi(p^k)$ are defined as follows,

$$d_\Phi(p^k) = \frac{|p^0 - p^k|}{\sqrt{2\sigma_W^2}}, \tag{8}$$

$$d_\Psi(p^k) = \frac{(\delta_H^2 + \delta_V^2)^{1/2}}{\sqrt{2\sigma_W^2}}. \tag{9}$$

Parameters δ_H and δ_V denote the values of horizontal and vertical displacements, and they are calculated as follows,

$$\delta_H = \begin{cases} -1, & k = \{1, 7, 8\}, \\ 0, & k = \{2, 6\}, \\ 1, & k = \{3, 4, 5\}, \end{cases} \qquad \delta_V = \begin{cases} -1, & k = \{1, 2, 3\}, \\ 0, & k = \{4, 8\}, \\ 1, & k = \{5, 6, 7\}. \end{cases} \tag{10}$$

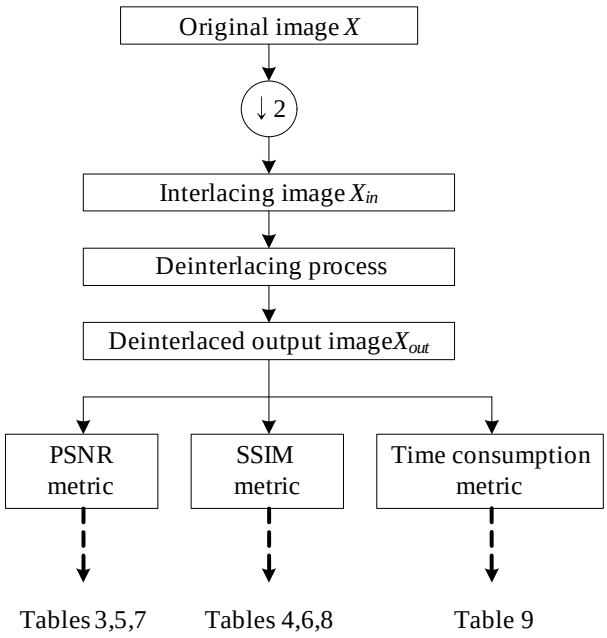


Figure 4: Block diagram of the objective performance evaluation process.

Fig. 3 shows the block diagram of the proposed algorithm.

3. Experimental Results

3.1. Experiments Setting

In this section, the performance of our method is evaluated and compared with some standard benchmarks. We used two objective metrics, peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) measure [21]. We also compared the consumed computational time. To assess performance, interlace process by splitting the odd and even numbered fields of an image was conducted, and then different deinterlacing methods were applied to restore the deinterlaced image. To validate the presented method, we conducted experiments using MATLAB with a 2.53GHz Intel(R) Core(TM) i5 CPU M460. The block diagram for the objective metrics is illustrated in Fig. 4.

Our method is compared with existing benchmarks including EMD, LCID, LABI, FEPD, MCAD, LSMD, and FFD. As a training set, we used 150 LC images [22]. Fig. 5 shows 25 selected training images. To test performance

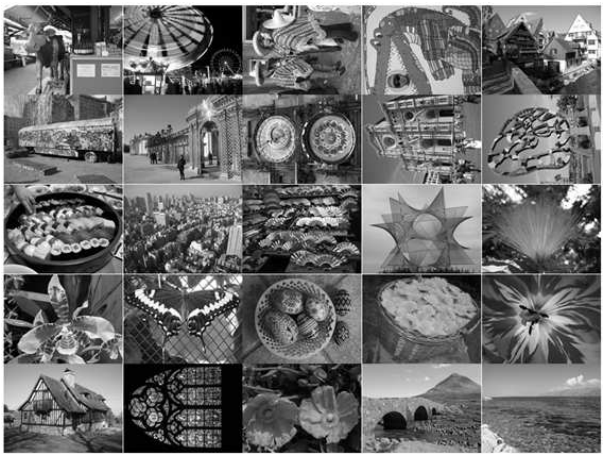


Figure 5: Training set: 25 selected images out of 150 LC database for training.

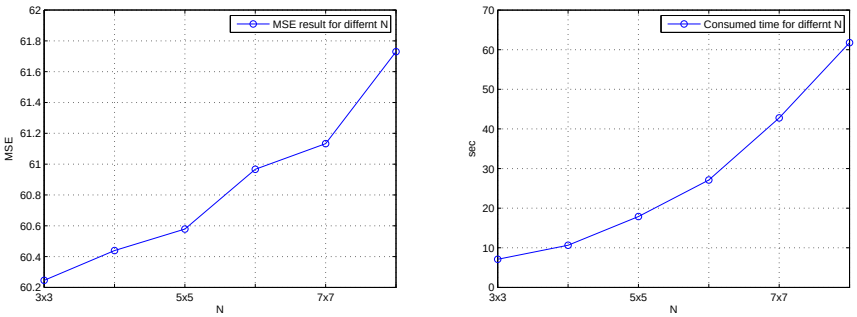
(a)

(b) (c)

Figure 6: Test sets: (a) Ten test images: AI, AK, BA, BL, BO, GI, FO, FI, CI, and BU (clockwise), (b) 18 McM database, (c) 25 Zahra database.

Table 1: Description of 10 standard images

	Resolution	Motion	Images
Test set	512×512	No	AI, BA, BO, FI, GI
	352×288	Yes	AK, BU, FO
	1920×1080	Yes	BL
	1280×720	Yes	CI



(a) (b)

Figure 7: (a) MSE result for varying N. (b) Consumed CPU time for varying N.

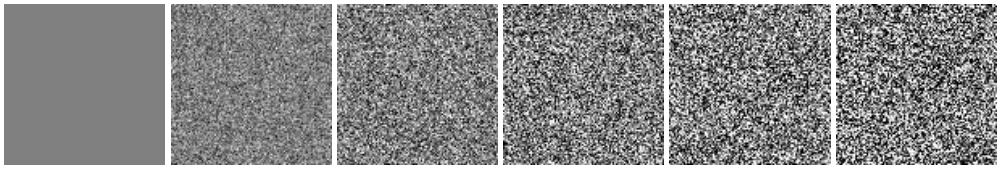
of the proposed method with benchmarks, we used three dataset: 10 standard images [23; 24], 18 McM dataset [25], and 25 Zahra dataset [26]. 10 standard images include Airplane (AI), Akiyo (AK), Barbara (BA), Bluesky (BL), Boat (BO), Bus (BU), City (CI), Finger (FI), Football (FO), and Girl (GI). All images are arranged in alphabetical order. Table 1 summarizes the information in terms of resolution and motion for 10 standard images. Fig. 6(a) shows test images listed in Table 1. The McM dataset and Zahra dataset are shown in Figs. 6(b) and 6(c).

3.2. Block Size Selection

To determine the best N, we tested MSE and CPU time performance by varying N (i.e., = 3, 5, 7, 9, 11, and 13) for $N \times N$ window. Table 2 and Fig. 7 show the MSE and average CPU time (sec) varying N for training images, under the condition of $\tau = 0$. We note that condition '$\tau = 0$' indicates BF is not used for the test, and all pixels are interpolated by FWAF. From

Table 2: Comparison of average MSE and CPU time (sec) of different N for 150 LC images, under the condition of $\tau = 0$.

NxN	3×3	5×5	7×7	9×9	11×11	13×13
MSE	60.2458	60.4392	60.5786	60.9666	61.1331	61.7297
CPU time	7.102	10.642	17.891	27.102	42.783	61.766



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

Figure 8: Generated images: (a) $(\mu, \sigma^2) = (0.5, 0.00)$, (b) $(\mu, \sigma^2) = (0.5, 0.02)$, (c) $(\mu, \sigma^2) = (0.5, 0.05)$, (d) $(\mu, \sigma^2) = (0.5, 0.10)$, (e) $(\mu, \sigma^2) = (0.5, 0.15)$, and (f) $(\mu, \sigma^2) = (0.5, 0.20)$.

the results, $N = 3$ was selected for the simulations of the paper as the best window size which provides the least MSE and requires the least CPU time (i.e., 60.25 of MSE and 7.5 sec of consumed time, respectively).

We note that this result seems to be caused by locality property. In fact, it is obvious that as the window size increases, the chance of similarity between corner and center pixels decreases while complexity increases.

3.3. Parameter Tuning

An appropriate settings for threshold parameter τ is important for the effectiveness of our method. Parameter τ plays an important role to balance performance and complexity. In our hybrid scheme, FWAF filter is applied in W_{HE} while a conventional BF is used in W_{HO}. It is obvious that the more we use the BF, the more image quality degrades, with the benefit of low complexity. On the other hand, the more we use the FWAF, the more image quality improves, but computational time increases as well. Therefore, our method is a hybrid one, which solves the rate-distortion optimization issues to improve deinterlacing quality under some constraints. To evaluate the threshold parameter τ, we used MSE metric for images and CPU time with different variances.

(a) (b)

Figure 9: Deinterlaced result of #1 LC image using (a) BF, (b) FWAF.

We generated artificial images using random distribution with a specific mean and variance values. More in details, $\mu = 0.5$ was used as mean value and six σ^2 values (= 0.00, 0.02, 0.05, 0.10, 0.15, and 0.20) were used as variance values, with the scope of tuning the parameter τ under different noise condition (see Fig. 8).

Fig. 9 shows deinterlaced #1 LC image with BF and FWAF. Fig. 10 shows μ_W and σ_W^2 maps and their histograms. As μ_W map is obtained by average values of window W, it looks similar to blurred image (Fig. 10a). From Fig. 10(b), it is obvious that σ_W^2 has high value in the heterogeneous area while σ_W^2 has low value in the homogeneous area. Figs. 10(c) and 10(d) show histograms of μ_W and σ_W^2 maps. It is noted that the average value of σ_W^2 for #1 LC image was 0.0036 which means most pixels have low σ_W^2.

We need to find a suitable parameter τ to adaptively balance the use of FWAF in W_{HE} and BF in W_{HO}. Fig. 11 shows MSE performance with different τ values using Eq. (5) and their corresponding CPU time. According to Eq. (5), FWAF is used when σ_W^2 is bigger than τ, otherwise BF is used. Fig. 11 shows the MSE and CPU time results for 150 LC images using above rule. From Fig. 11, we empirically choose $\tau = 0.062$ providing reliable performance and a good quality-speed tradeoff among all τ parameters. It also can be find that only 0.3407% (1,325 out of 388,800 pixels) of #1 LC image pixels show $\sigma_W^2 > 0.062$. Therefore, the complexity of FWAF becomes low.

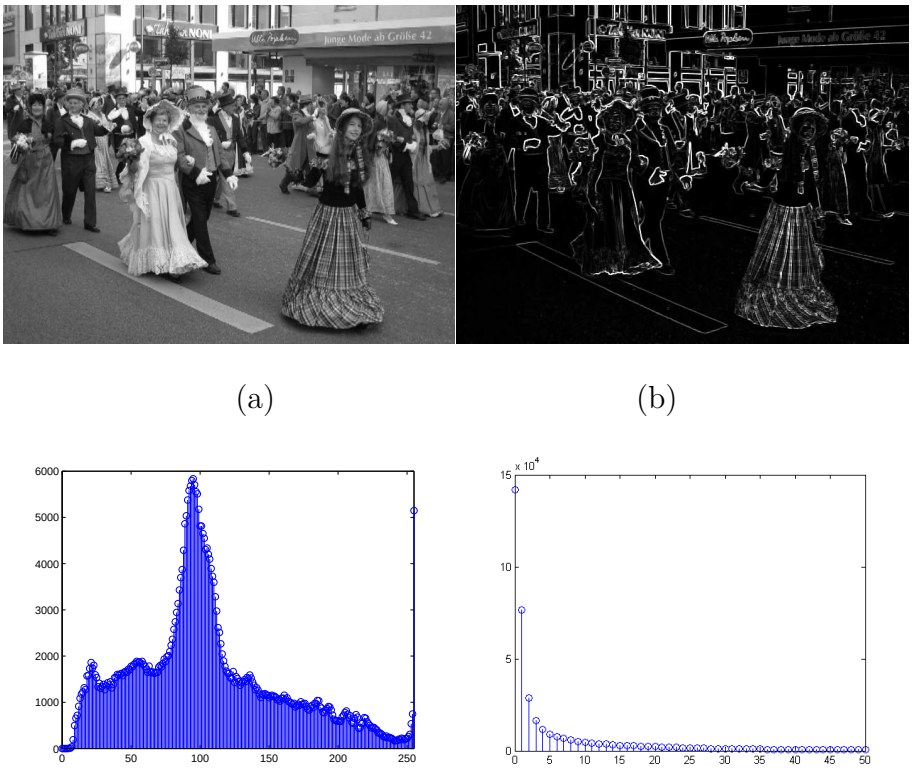
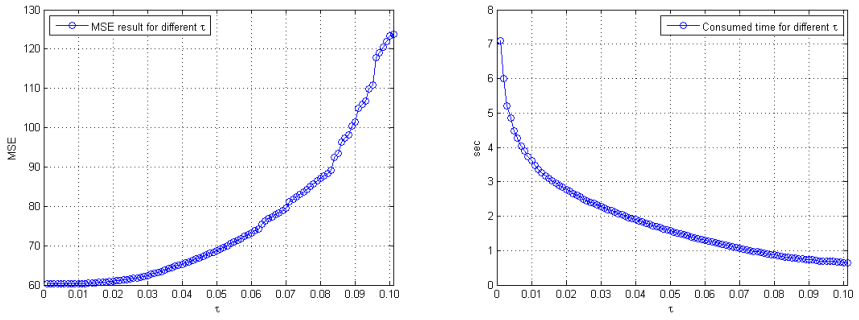


Figure 10: Two maps of μ_W and σ_W^2 and their histograms for #1 LC image. As σ_W^2 is very small values, 20 was multiplied to clearly represent σ_W^2: (a) μ_W map, (b) σ_W^2 map, (c) μ_W histogram, and (d) σ_W^2 histogram.



(a) (b)

Figure 11: MSE results for 150 LC dataset: (a) MSE result for varying τ, (b) CPU time for varying τ. We then select $\tau = 0.062$ as a good tradeoff for our system.

3.4. Objective Performance Comparison

To assess objective performance of our method with conventional benchmarks, we used two metrics. One is peak signal-to-noise ratio (PSNR) and the other is the structural similarity (SSIM). The PSNR metric is defined as

$$MSE(img_{org}, img_{rec}) = \sum_{p=1}^{width} \sum_{q=1}^{height} \frac{(img_{org}(p,q) - img_{rec}(p,q))^2}{width \times height}, \tag{11}$$

$$PSNR(img_{org}, img_{rec}) = 10 \log_{10} \frac{255^2}{MSE(img_{org}, img_{rec})}, \tag{12}$$

where img_{org} and img_{rec} are the *original* and *reconstructed* images, respectively. All the test images were converted from the original size into the vertically interlaced size, and then the reconstructed images, img_{rec}, were compared to the original image, img_{org}.

In order to consistently evaluate image quality, we also apply a measurement called SSIM. The SSIM uses a perceptual model in an attempt to measure the subjective performance [21]. A smaller SSIM value denotes more error in the estimated image and hence poorer perceived visual quality. The SSIM index compares local patterns of pixel intensities normalized for luminance and contrast, and we can measure the similarity between the original image and the reconstructed image via SSIM. The focus of the SSIM index is to capture the loss of structure in images. Since the human visual

system is highly adaptable, it is able to extract structural information from a visual scene. Therefore, a measurement of structural similarity provides a good approximation of perceived image quality. The SSIM index is denoted by

$$SSIM(img_{org}, img_{rec}) = \frac{(2\mu_O\mu_R + T_1)(2\sigma_{OR} + T_2)}{(\mu_O^2 + \mu_R^2 + T_1)(\sigma_O^2 + \sigma_R^2 + T_2)}, \tag{13}$$

where σ_{OR} indicates the covariance of the two local regions of img_{org} and img_{rec}, and μ_O (or σ_O) and μ_R (or σ_R) represent the mean (or standard deviation) of the specific and local regions, respectively. The two constants T_1 and T_2 are used to avoid instability when $\mu_O^2+\mu_R^2$ or $\sigma_O^2+\sigma_R^2$ is approaching zero. The mean SSIM index is adopted to evaluate the overall image quality and is given by

$$MSSIM(img_{org}, img_{rec}) = \frac{1}{M}\sum_{k=1}^{M} SSIM(img_{org,k}, img_{rec,k}), \tag{14}$$

where M is the number of local image regions. We discover from Eq. (13) that, when μ_O is equal to μ_R and σ_O is equal to σ_R, SSIM is the largest at a value of 1. Thus, the SSIM ranges from 0 to 1. If the SSIM is closer to 1, then the reconstructed image is more similar to the original image, and vice versa. The MSSIM is the mean of the SSIM; thus, the MSSIM has the same properties as the SSIM. Namely, the larger is the MSSIM, the better is the subjective image quality.

Table 3 and Table 4 show PSNR and SSIM performance comparisons, respectively. A term 'avg.' stands for the average results of each column. Two terms 'R_τ' and 'R_0' stand for the ranking of $FWAF_\tau$ and $FWAF_0$, respectively. Here, $FWAF_\tau$ and $FWAF_0$ are results obtained when $\tau = 0.062$ and $\tau = 0$, respectively. A term 'R_M' stands for the ranking of each method in terms of PSNR or SSIM. Table 3 shows that the $FWAF_0$ (or $FWAF_\tau$) achieved 0.276 (or 0.048) dB better than the best benchmark, MELA. Table 3 also shows that ranking of $FWAF_\tau$ ranges from 2 to 6. For BO image, ranking of $FWAF_\tau$ was 6. However, the ranking of average PSNR is 2.

Although a higher PSNR usually implies that the restoration is of higher quality, experience has shown that sometimes this turns out not to be true. In order to better assess the objective performance of the presented method, we used another metric of structural similarity, SSIM. The SSIM metric considers the local patterns of pixel values which were standardized for luminance

Table 3: Performance comparison with PSNR metric (dB) for 10 standard images: R_τ and R_0 denote ranks of $FWAF_\tau$ and $FWAF_0$ of each image, and R_M denotes rank of each method

	EMD	LCID	MELA	LABI	FEPD	MCAD	LSMD	FFD	$FWAF_\tau$	$FWAF_0$	R_τ	R_0
AI	34.659	35.022	35.088	35.345	34.385	35.085	35.66	35.118	35.177	35.423	4	2
AK	38.457	39.882	40.205	38.841	37.255	39.726	38.149	39.692	40.166	40.557	3	1
BA	30.175	31.632	32.018	31.93	28.879	25.929	29.414	31.935	31.947	31.949	3	2
BL	37.628	37.913	37.9	37.798	37.51	38.107	39.373	37.792	38.029	38.456	4	2
BO	34.002	34.908	35.186	35.277	33.074	35.342	33.762	35.178	35.169	35.314	6	2
BU	28.453	28.615	28.654	28.217	28.104	28.262	28.095	28.507	28.757	28.772	2	1
CI	31.343	31.444	31.46	31.497	31.258	31.527	31.656	31.444	31.574	31.58	3	2
FI	31.064	31.223	31.323	31.362	30.679	31.81	32.085	31.33	31.395	31.863	4	2
FO	34.211	34.972	35.057	34.475	33.308	35.034	34.763	34.813	35.137	35.333	2	1
GI	40.758	41.607	41.793	41.535	39.676	42.038	41.545	41.651	41.822	42.198	3	1
avg.	34.075	34.722	34.868	34.628	33.413	34.286	34.45	34.746	34.917	35.144	2	1
R_M	9	5	3	6	10	8	7	4	2	1		

Table 4: Performance comparison with SSIM metric for 10 standard images: R_τ and R_0 denote ranks of $FWAF_\tau$ and $FWAF_0$ of each image, and R_M denotes rank of each method

	EMD	LCID	MELA	LABI	FEPD	MCAD	LSMD	FFD	$FWAF_\tau$	$FWAF_0$	R_τ	R_0
AI	0.9658	0.9673	0.9677	0.9686	0.9649	0.9665	0.9685	0.9691	0.9685	0.9686	4	2
AK	0.9857	0.9883	0.989	0.9866	0.9823	0.986	0.9839	0.9875	0.9897	0.9898	2	1
BA	0.9347	0.9463	0.9487	0.9479	0.9227	0.8952	0.9311	0.9409	0.9485	0.9486	3	2
BL	0.9805	0.9814	0.9812	0.9809	0.981	0.9821	0.9844	0.9838	0.9823	0.9825	4	3
BO	0.933	0.9372	0.9375	0.9375	0.9315	0.9375	0.9361	0.9378	0.9384	0.9385	2	1
BU	0.9151	0.9186	0.9197	0.9162	0.9073	0.9008	0.9094	0.9156	0.9202	0.9204	2	1
CI	0.9152	0.9179	0.9179	0.9184	0.9129	0.9186	0.9184	0.9192	0.9189	0.919	3	2
FI	0.9519	0.9535	0.9546	0.9558	0.9471	0.9588	0.9608	0.9587	0.9551	0.9601	6	2
FO	0.9487	0.9516	0.951	0.9464	0.9449	0.9476	0.9499	0.9515	0.9523	0.9526	2	1
GI	0.9786	0.9813	0.982	0.9825	0.9748	0.9823	0.9809	0.9825	0.9827	0.9829	2	1
avg.	0.9509	0.9543	0.9549	0.9541	0.9469	0.9475	0.9523	0.9547	0.9557	0.9563	2	1
R_M	8	5	3	6	10	9	7	4	2	1		

Table 5: Performance comparison with PSNR metric (dB) for 18 McM dataset: R_τ and R_0 denote ranks of $FWAF_\tau$ and $FWAF_0$ of each image, and R_M denotes rank of each method

	EMD	LCID	MELA	LABI	FEPD	MCAD	LSMD	FFD	$FWAF_\tau$	$FWAF_0$	R_τ	R_0
1	29.638	30.25	30.326	30.211	29.219	29.797	29.708	30.30601	30.282	30.417	4	1
2	33.341	33.959	34.04	33.873	32.844	33.42	33.433	34.020212	33.964	34.269	4	1
3	28.746	29.217	29.237	29.236	28.326	28.824	28.987	29.216653	29.344	29.482	2	1
4	31.471	32.534	32.676	31.95	30.524	31.205	32.062	32.655604	32.39	33.166	5	1
5	35.07	35.64	35.67	35.513	34.672	35.163	35.637	35.650172	35.821	35.888	2	1
6	38.565	39.103	39.203	39.287	38.045	38.683	39.922	39.182785	39.547	39.948	3	1
7	31.455	31.711	31.712	31.794	31.472	31.778	31.732	31.692013	31.871	31.917	2	1
8	32.532	32.904	33.001	33.189	32.168	32.786	32.983	32.981497	33.085	33.365	3	1
9	34.735	35.474	35.521	35.184	34.281	34.797	35.581	35.501021	35.65	35.757	2	1
10	37.647	38.301	38.375	37.852	37.146	37.404	38.219	38.355021	38.328	38.725	4	1
11	37.904	38.394	38.452	38.233	37.611	37.947	38.483	38.431934	38.64	38.657	2	1
12	36.231	37.369	37.552	37.459	35.038	36.134	36.332	37.532452	37.527	37.887	4	1
13	39.924	40.786	40.846	40.561	39.208	39.952	40.629	40.826201	40.99	41.21	2	1
14	38.296	38.835	38.889	38.524	37.897	38.271	38.584	38.868657	38.721	39.15	5	1
15	39.766	40.528	40.622	39.89	39.141	39.473	40.029	40.601845	40.162	40.785	5	1
16	30.731	31.161	31.207	31.07	30.493	30.786	31.321	31.18736	31.347	31.422	2	1
17	34.96	35.459	35.511	35.37	34.672	35.027	35.704	35.491029	35.639	35.789	3	1
18	30.268	30.613	30.665	30.542	29.988	30.309	30.562	30.645183	30.823	31.129	2	1
avg.	34.516	35.124	35.195	34.985	34.041	34.542	34.995	35.175	35.23	35.498	2	1
R_M	9	5	3	7	10	8	6	4	2	1		

and contrast. We measured the similarity between two images (original progressive image and deinterlaced image) by utilizing SSIM. We note that when SSIM value appeal to 1, the deinterlaced image becomes more similar to the original one. Table 4 shows SSIM performance. It can be found that the proposed $FWAF_0$ and $FWAF_\tau$ methods outperformed the best benchmark, MELA, by 0.0014 and 0.0008, in terms of SSIM. It is clear that $FWAF_0$ and $FWAF_\tau$ are not efficient for FI or BL images. However, in general, $FWAF_0$ and $FWAF_\tau$ methods ranked in #1 and #2, respectively.

Tables 5 and 6 show PSNR and SSIM results for 18 McM dataset, and Tables 7 and 8 show PSNR and SSIM results for 25 Zahra dataset. Table 5 shows that $FWAF_0$ and $FWAF_\tau$ provide the best and the second best solutions in terms of PSNR metric, and MELA and FFD are ranked in #3 and #4. Quite similar results can be seen in Table 6 where $FWAF_0$ and $FWAF_\tau$ still provided the best and the second best SSIM results. We not that PSNR

Table 6: Performance comparison with SSIM metric for 18 McM dataset: R_τ and R_0 denote ranks of $FWAF_\tau$ and $FWAF_0$ of each image, and R_M denotes rank of each method

	EMD	LCID	MELA	LABI	FEPD	MCAD	LSMD	FFD	$FWAF_\tau$	$FWAF_0$	R_τ	R_0
1	0.9226	0.9308	0.9315	0.9319	0.917	0.9268	0.922	0.9315	0.932	0.9323	2	1
2	0.9488	0.9538	0.9543	0.9541	0.9454	0.9505	0.9487	0.9543	0.9544	0.9549	2	1
3	0.9333	0.9392	0.9393	0.9392	0.9277	0.9344	0.9352	0.9393	0.9402	0.9412	2	1
4	0.973	0.978	0.9785	0.9762	0.9672	0.9722	0.9752	0.9784	0.9777	0.9799	5	1
5	0.9588	0.9621	0.9623	0.9623	0.9569	0.9605	0.9627	0.9622	0.9632	0.9642	2	1
6	0.9747	0.9767	0.9771	0.978	0.9726	0.9756	0.9796	0.9771	0.9779	0.978	4	2
7	0.913	0.9165	0.9168	0.9182	0.9128	0.9173	0.915	0.9168	0.9168	0.9179	4	2
8	0.9617	0.9641	0.9645	0.9645	0.9596	0.9626	0.9626	0.9644	0.9639	0.9642	6	4
9	0.966	0.9698	0.97	0.969	0.9634	0.9669	0.9694	0.97	0.9705	0.9709	2	1
10	0.9701	0.9733	0.9736	0.972	0.9676	0.9697	0.9725	0.9736	0.9739	0.9741	2	1
11	0.9678	0.9704	0.9708	0.9705	0.9661	0.9687	0.9708	0.9708	0.9716	0.9722	2	1
12	0.9744	0.9781	0.9785	0.9784	0.9698	0.9748	0.9744	0.9785	0.9786	0.9787	2	1
13	0.978	0.9799	0.98	0.9797	0.9772	0.979	0.9787	0.9799	0.9792	0.9799	6	4
14	0.9674	0.969	0.9693	0.9687	0.9667	0.9682	0.9682	0.9693	0.9694	0.9697	2	1
15	0.9693	0.9714	0.9716	0.9703	0.9682	0.9693	0.9695	0.9716	0.971	0.9716	5	1
16	0.9299	0.9352	0.9357	0.935	0.9261	0.9307	0.937	0.9357	0.9369	0.939	3	1
17	0.9536	0.958	0.9584	0.9571	0.9508	0.954	0.9593	0.9584	0.959	0.9604	3	1
18	0.9294	0.9339	0.9348	0.9365	0.9235	0.9314	0.9283	0.9348	0.935	0.9369	3	1
avg.	0.9551	0.9589	0.9593	0.959	0.9521	0.9563	0.9572	0.9593	0.9595	0.9603	2	1
R_M	9	6	3	5	10	8	7	4	2	1		

Table 7: Performance comparison with PSNR metric (dB) for 25 Zahra dataset: R_τ and R_0 denote ranks of $FWAF_\tau$ and $FWAF_0$ of each image, and R_M denotes rank of each method

	EMD	LCID	MELA	LABI	FEPD	MCAD	LSMD	FFD	$FWAF_\tau$	$FWAF_0$	R_τ	R_0
1	27.513	27.764	27.776	27.691	27.437	27.63	27.496	27.771	27.769	27.798	4	1
2	33.941	34.248	34.282	34.495	33.623	34.147	34.392	34.277	34.493	34.673	3	1
3	30.474	30.946	31.005	30.902	30.024	30.551	30.359	31	30.95	31.098	4	1
4	35.053	35.375	35.479	35.748	34.648	35.282	35.354	35.474	35.687	35.95	3	1
5	29.261	29.457	29.458	29.504	29.325	29.512	29.486	29.453	29.52	29.636	2	1
6	35.638	35.832	35.838	35.974	35.572	35.875	35.938	35.833	35.846	36.125	5	1
7	32.104	32.85	32.947	33.147	31.141	32.259	31.815	32.942	33.01	33.192	3	1
8	33.932	34.294	34.346	34.407	33.499	34.062	33.764	34.341	34.355	34.485	3	1
9	32.941	33.093	33.146	33.386	32.756	33.136	33.367	33.141	33.337	33.601	4	1
10	29.006	29.277	29.334	29.364	28.788	29.111	28.915	29.329	29.36	29.385	3	1
11	31.061	31.27	31.282	31.226	31.07	31.183	31.247	31.277	31.322	31.402	2	1
12	35.688	35.866	35.861	35.68	35.731	35.711	35.816	35.856	35.853	35.897	5	1
13	27.574	28.363	28.564	28.703	26.405	27.517	27.002	28.559	28.427	28.844	5	1
14	28.617	28.855	28.883	28.795	28.501	28.704	28.478	28.878	28.886	28.894	2	1
15	31.898	32.282	32.321	32.347	31.162	31.902	31.379	32.316	32.409	32.484	2	1
16	31.028	31.507	31.563	31.327	30.849	31.145	30.619	31.558	31.325	31.599	6	1
17	33.454	33.848	33.861	33.753	33.397	33.609	33.811	33.856	33.953	34.019	2	1
18	34.46	34.677	34.663	34.651	34.54	34.72	34.458	34.658	34.675	34.698	4	2
19	31.802	32.06	32.113	32.034	31.556	31.844	31.566	32.108	32.035	32.187	5	1
20	38.654	38.793	38.755	38.951	38.31	38.749	38.468	38.75	38.817	38.963	3	1
21	39.662	40.577	40.787	40.97	38.968	40.064	39.624	40.782	40.592	41.063	5	1
22	34.26	34.653	34.517	34.463	34.286	34.516	33.891	34.512	34.599	34.634	3	2
23	34.209	34.789	34.821	34.57	33.963	34.441	34.235	34.816	34.712	34.901	5	1
24	26.879	27.086	27.006	27.108	26.245	26.708	27.031	27.001	27.103	27.394	3	1
25	35.077	35.949	36.059	35.63	34.639	35.265	36.022	36.054	36.112	36.661	2	1
avg.	32.567	32.948	32.987	32.993	32.257	32.706	32.581	32.982	33.006	33.183	2	1
R_M	9	6	4	3	10	7	8	5	2	1		

Table 8: Performance comparison with SSIM metric for 25 Zahra dataset: R_τ and R_0 denote ranks of $FWAF_\tau$ and $FWAF_0$ of each image, and R_M denotes rank of each method

	EMD	LCID	MELA	LABI	FEPD	MCAD	LSMD	FFD	$FWAF_\tau$	$FWAF_0$	R_τ	R_0
1	0.8904	0.8953	0.8956	0.8959	0.8888	0.8942	0.8907	0.8956	0.8955	0.8969	5	1
2	0.9445	0.9472	0.9474	0.949	0.9416	0.9466	0.9465	0.9474	0.9485	0.9498	3	1
3	0.9284	0.9338	0.9343	0.9344	0.9231	0.9307	0.928	0.9343	0.9349	0.935	2	1
4	0.9716	0.9729	0.9733	0.9747	0.9699	0.9729	0.9734	0.9733	0.974	0.9758	3	1
5	0.8817	0.8862	0.8868	0.8905	0.8814	0.8881	0.8858	0.8868	0.888	0.8921	4	1
6	0.9528	0.9542	0.9542	0.9546	0.9527	0.9542	0.9546	0.9542	0.9538	0.956	8	1
7	0.9566	0.9617	0.9623	0.9634	0.949	0.958	0.9553	0.9623	0.9632	0.9638	3	1
8	0.9422	0.9467	0.9471	0.9479	0.9376	0.9448	0.941	0.947	0.9475	0.9476	3	2
9	0.9394	0.9411	0.9417	0.9443	0.9376	0.9413	0.9425	0.9417	0.9428	0.9457	3	1
10	0.8907	0.8947	0.8954	0.8964	0.8885	0.8932	0.8883	0.8954	0.8955	0.8957	3	2
11	0.9053	0.9086	0.9091	0.9106	0.9048	0.9087	0.9086	0.909	0.9098	0.9128	3	1
12	0.9373	0.9385	0.9385	0.9379	0.9381	0.9383	0.939	0.9385	0.9385	0.9397	6	1
13	0.9135	0.9212	0.9227	0.9246	0.8988	0.9133	0.9056	0.9226	0.9223	0.9225	5	4
14	0.9055	0.91	0.9103	0.9102	0.9027	0.9077	0.9022	0.9103	0.9094	0.9101	6	4
15	0.9601	0.9629	0.963	0.9618	0.9549	0.9593	0.9551	0.963	0.9617	0.9619	6	4
16	0.9448	0.9504	0.9508	0.9494	0.942	0.9464	0.9385	0.9508	0.9481	0.9504	6	4
17	0.9628	0.9651	0.9652	0.965	0.9624	0.9639	0.9649	0.9651	0.9651	0.9666	5	1
18	0.9314	0.9343	0.934	0.9343	0.9327	0.9354	0.9308	0.934	0.934	0.9345	5	2
19	0.923	0.9274	0.928	0.9267	0.9197	0.9241	0.9181	0.928	0.9282	0.9285	2	1
20	0.9798	0.9804	0.9802	0.9802	0.9797	0.9805	0.9793	0.9801	0.9802	0.9806	4	1
21	0.9852	0.9877	0.9882	0.9887	0.9831	0.9864	0.9853	0.9882	0.9888	0.9888	2	1
22	0.9666	0.969	0.9679	0.9672	0.967	0.9683	0.964	0.9679	0.968	0.968	4	3
23	0.9247	0.9284	0.9279	0.9275	0.9264	0.93	0.9243	0.9278	0.9268	0.9283	7	3
24	0.9418	0.9434	0.9427	0.9426	0.9376	0.9414	0.9407	0.9427	0.9438	0.9451	2	1
25	0.9752	0.9784	0.9788	0.9781	0.973	0.9761	0.9795	0.9787	0.9804	0.9804	2	1
avg.	0.9382	0.9416	0.9418	0.9422	0.9357	0.9402	0.9377	0.9418	0.9419	0.9431	3	1
R_M	8	6	4	2	10	7	9	5	3	1		

Table 9: Consumed average processing time (sec): R_τ and R_0 denote ranks of $FWAF_\tau$ and $FWAF_0$ of each image, and R_M denotes rank of each method

	EMD	LCID	MELA	LABI	FEPD	MCAD	LSMD	FFD	$FWAF_\tau$	$FWAF_0$	R_τ	R_0
Standard	5.047	1.399	0.735	25.541	51.498	36.113	7.808	0.441	0.422	7.852	1	7
McM	4.001	1.277	0.781	19.303	38.684	27.197	6.062	0.561	0.547	6.095	1	7
Zahra	6.159	1.875	1.095	30.223	60.703	42.637	9.401	0.75	0.728	9.452	1	7
avg.	5.069	1.517	0.87	25.022	50.295	35.316	7.757	0.584	0.566	7.8	1	7
R_M	5	4	3	8	10	9	6	2	1	7		

performance of $FWAF_0$ was always the best for every 18 image. However, PSNR performance of $FWAF_\tau$ ranged from 2 to 5, and MELA and FFD were good competitors.

In terms of SSIM results, although the $FWAF_0$ and $FWAF_\tau$ provided the best and the second best performance in average, however they failed to give good results for some images such as #8 and #13. The reason is that, most pixels of #8 and #13 images of McM dataset are in a flat region, and both images have many lines with distinct edge direction. Therefore, edge direction based method may restore better image.

Table 7 shows PSNR results of $FWAF_0$ and $FWAF_\tau$, where both methods provided the best and the second best results. For two images (#18 and #22), $FWAF_0$ was not the best. However, the difference from the best method was negligible (0.022 from MCAD, 0.019 from LCID). In the same manner, $FWAF_0$ shows the best performance in terms of SSIM as shown in Table 8, but the results of each image is not always the best.

The $FWAF_\tau$ method still shows comparable PSNR results. Although it does not show the second best PSNR for all images, but its average results eventually were ranked as #2. However, it is obvious that $FWAF_\tau$ is not the second best method, and LABI outperformed $FWAF_\tau$. It is interesting observation that normally LABI was not well used due to its high complexity. The LABI method decides the direction and slope of edge based on locally adaptive-thresholded binary image. We assume this property might improve efficiency and accuracy of restoration. Another observation is that LABI may increase the horizontal size of search window to 15, which may give good direction determination at gentle slope.

Table 9 provides the comparison of CPU processing time which reflects time complexity. Three terms 'Standard,' 'McM,' and 'Zahra' stand for used test images, and 'avg.' and 'R_M' are average results and ranking of each

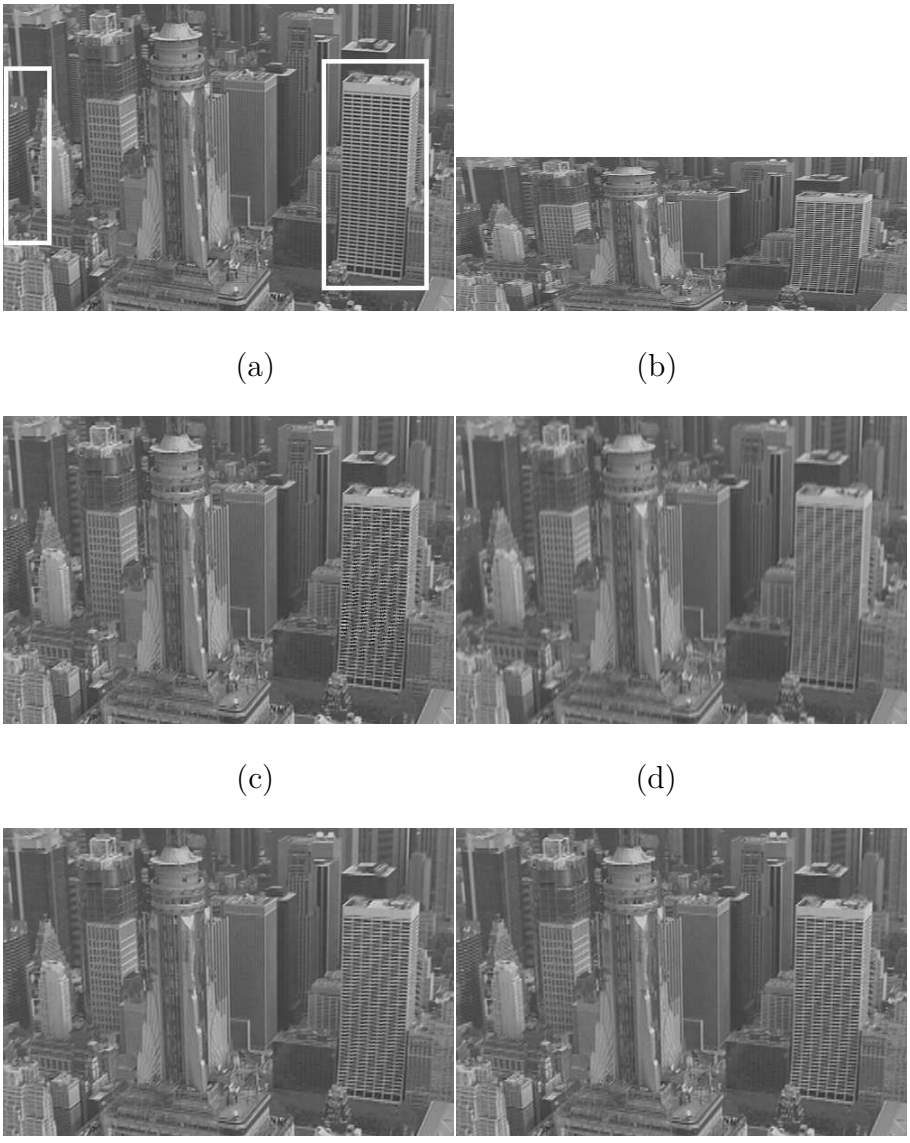
method. In the same manner, R_τ and R_0 denote ranks of $FWAF_\tau$ and $FWAF_0$ of each dataset. It is assumed that the method with smallest CPU time consumption is better than the other. It can be observed that FFD, MELA, and LCID methods requires small CPU time. However, $FWAF_\tau$ required only 95%, 57%, and 30% of the CPU time compared to FFD, MELA, and LCID, respectively. The FFD method uses pre-trained filter, which expedite deinterlacing implementation.

On the other hand, CPU time result of $FWAF_0$ was ranked #7, which can assumed as slow method. Although it has strength in terms of PSNR and SSIM, however it cannot be recommended for real-time system. Therefore, $FWAF_\tau$ method with appropriate threshold τ would be recommended.

3.5. Subjective Performance Comparison

Figs. 12 and 13 exhibit the edge protecting performance using a 'City' image. As MELA and FFD outperformed the other conventional methods in terms of PSNR and SSIM, these two methods were compared with the proposed method. By comparing the proposed method with MELA and FFD, we could observe that the presented approach has favorable visual result. From images shown in Figs. 12(e,f) and 13(e,f), it is clearly seen that the edge of buildings obtained with our method is sharper than the edges of other methods and there were no apparent artifacts in the homogeneous area. When comparing the marked parts, the proposed method performed better than the other methods. MELA had advantages in reconstruction of the sharp edge. However, it was ineffective when dealing with complex edges in heterogeneous area. It can be observed that abruptly interpolated values were shown in Figs. 12(b) and 13(b). FFD shows favorable reconstructed images, however some edges were blurred after deinterlacing because the same trained filter was applied to the whole image. On the other hand, reconstructed images using our method provided the best visual performance.

The simulation was also tested in three McM dataset (#1, #4, and #8) and three Zahra dataset (#1, #4, and #24). To allow visual quality evaluation, Figs. 14-19 show details of original part of test image and the deinterlaced outputs from the different conventional deinterlacing methods. MELA gives satisfactory results especially for the diagonal edges. However, there are severe drawbacks of MELA in the detailed diagonal edge areas (the edge direction is inconsistent). FFD shows quite good visual quality in most of patterns. However as FFD is based on pre-determined filter on training set,



(a) (b)

(c) (d)

(e) (f)

Figure 12: Comparison of the subjective performance of different deinterlacing methods with the proposed method: (a) Cropped original "City" image, (b) interlaced image, (c) MELA, (d) FFD, (e) FWAF_{τ} and (d) FWAF_{0}.

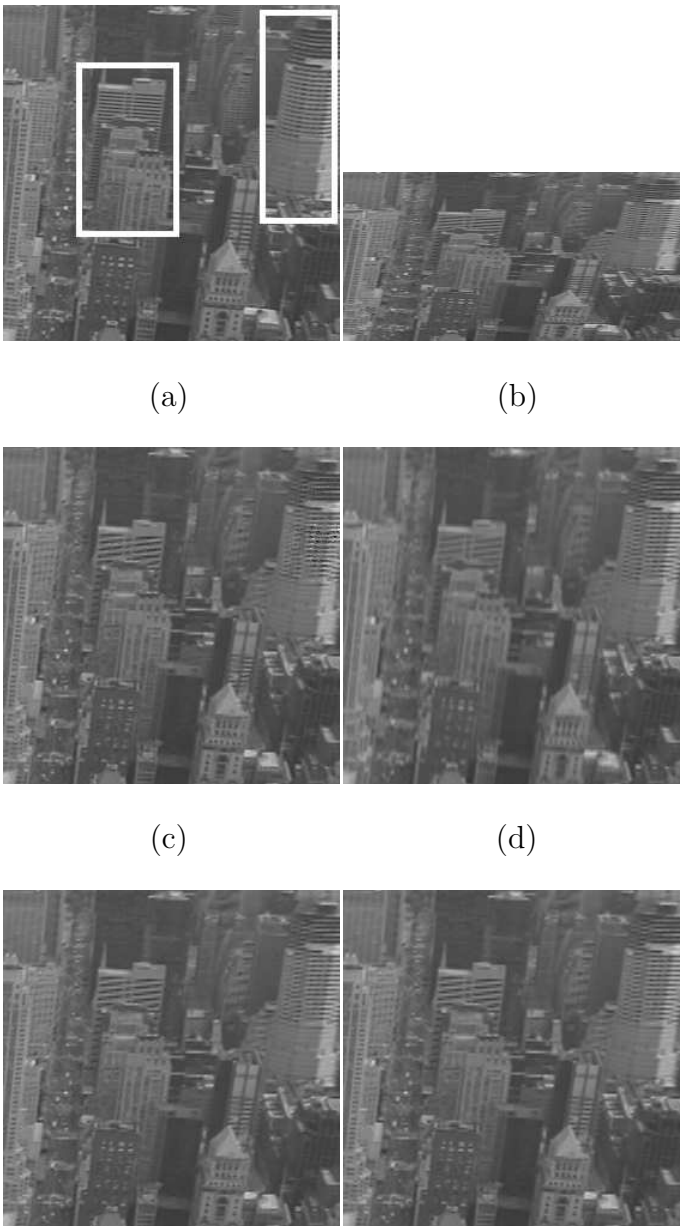
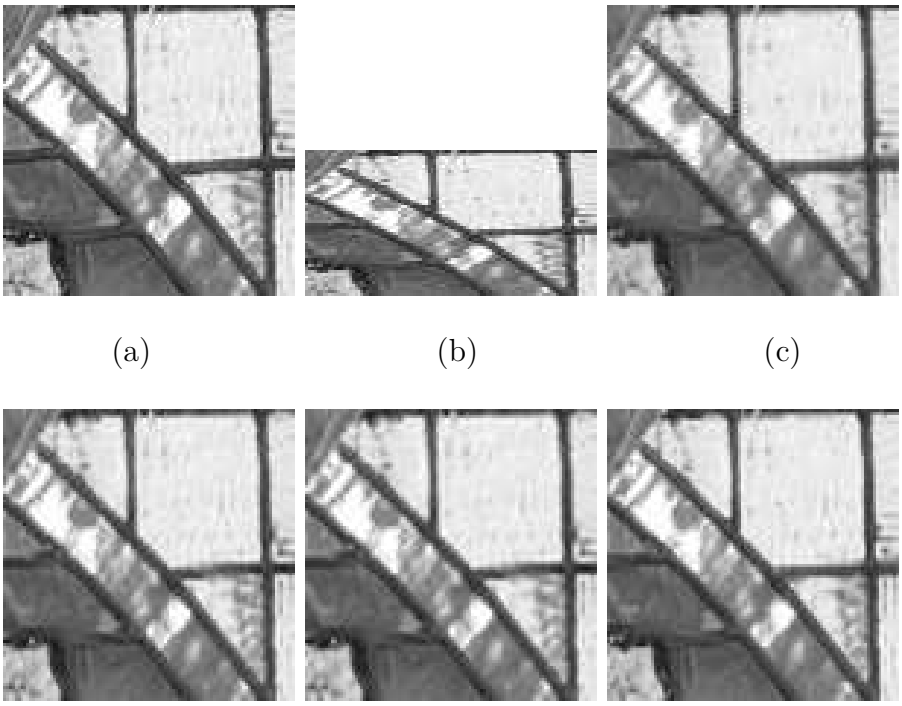


Figure 13: Comparison of the subjective performance of different deinterlacing methods with the proposed method: (a) Cropped original "City" image, (b) interlaced image, (c) MELA, (d) FFD, (e) FWAF_{τ} and (d) FWAF_{0}.



(a) (b) (c)

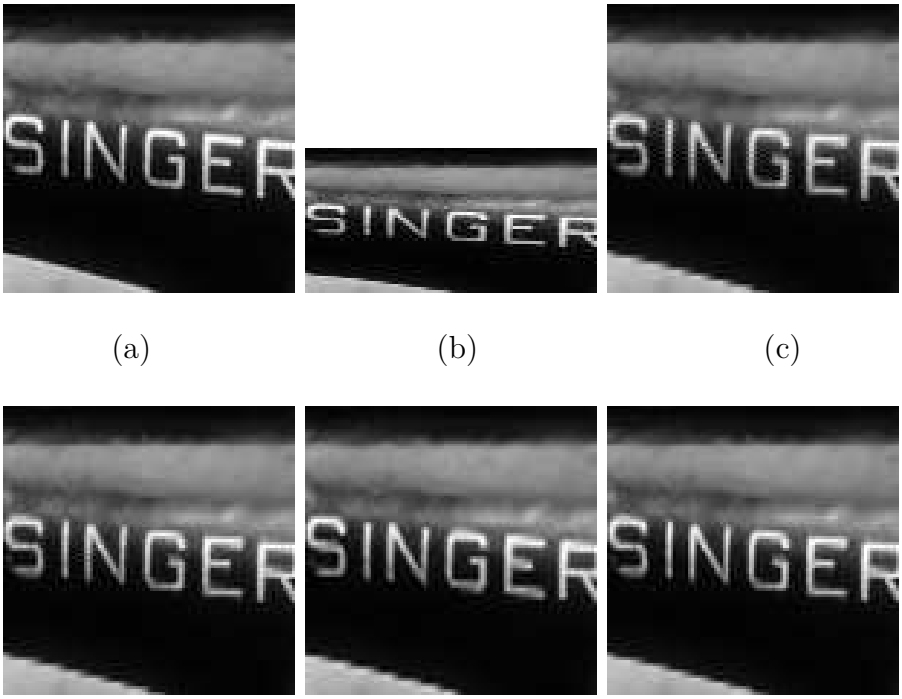
(d) (e) (f)

Figure 14: Comparison of the subjective performance of different deinterlacing methods with the proposed method: (a) Cropped original #1 McM image, (b) interlaced image, (c) MELA, (d) FFD, (e) FWAF_{τ} and (d) FWAF_{0}.

(a) (b) (c)

(d) (e) (f)

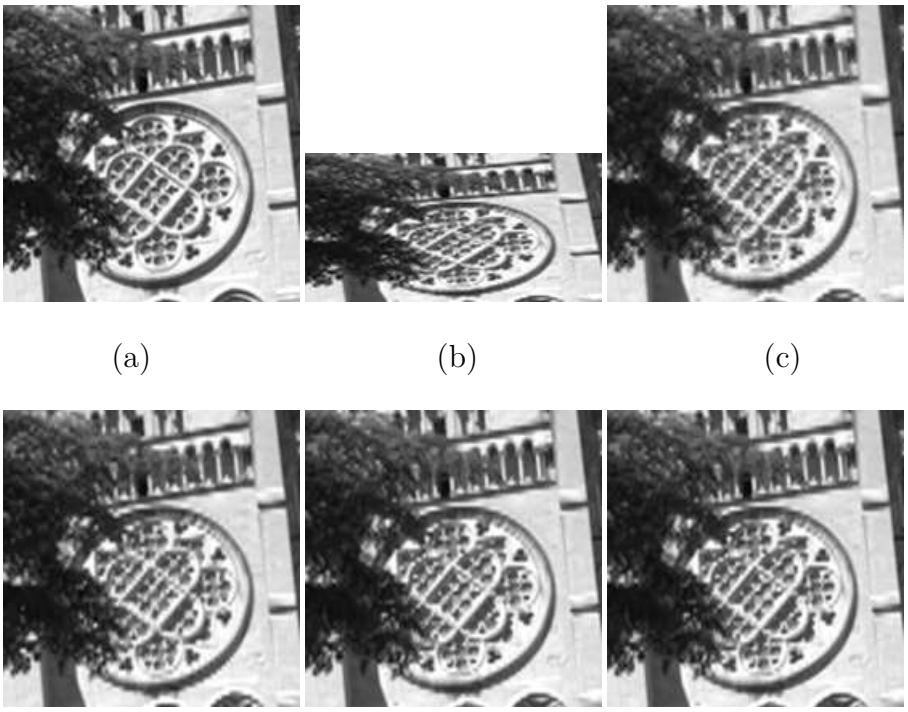
Figure 15: Comparison of the subjective performance of different deinterlacing methods with the proposed method: (a) Cropped original #4 McM image, (b) interlaced image, (c) MELA, (d) FFD, (e) FWAF_{τ} and (d) FWAF_{0}.



(a) (b) (c)

(d) (e) (f)

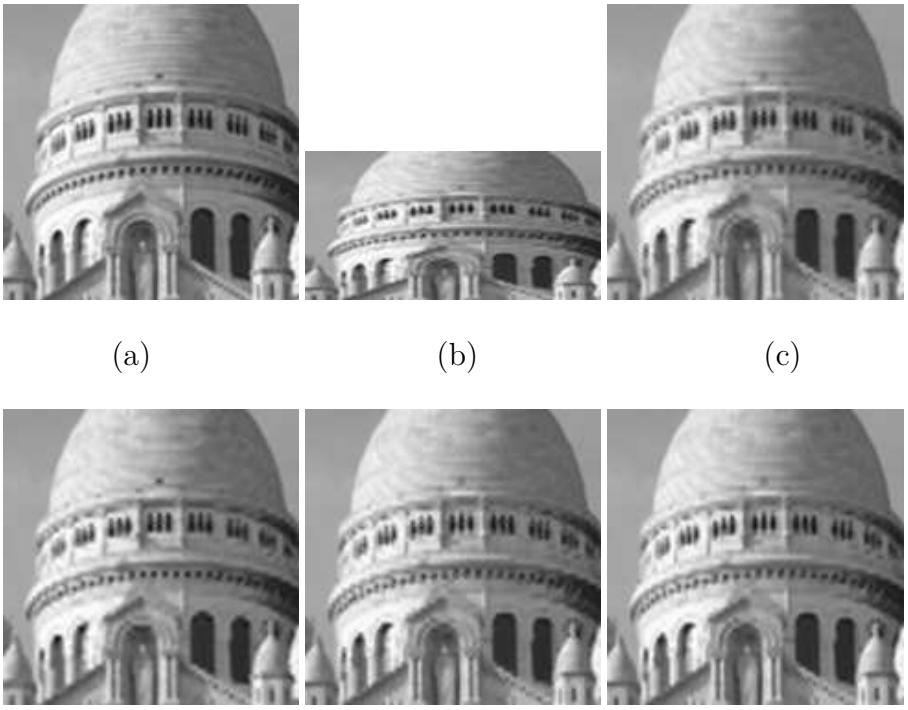
Figure 16: Comparison of the subjective performance of different deinterlacing methods with the proposed method: (a) Cropped original #8 McM image, (b) interlaced image, (c) MELA, (d) FFD, (e) FWAF_{τ} and (d) FWAF_{0}.



(a) (b) (c)

(d) (e) (f)

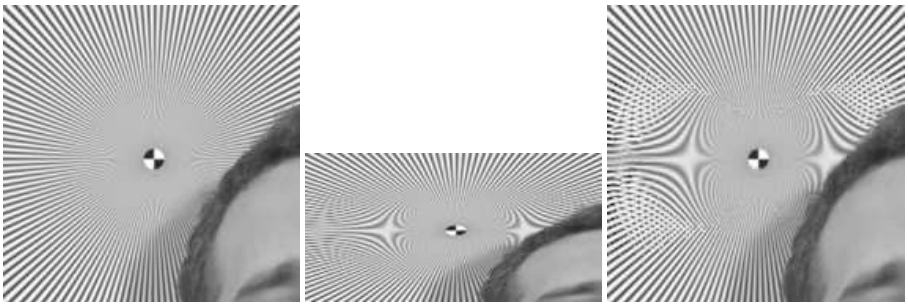
Figure 17: Comparison of the subjective performance of different deinterlacing methods with the proposed method: (a) Cropped original #1 Zahra image, (b) interlaced image, (c) MELA, (d) FFD, (e) FWAF_{τ} and (d) FWAF_{0}.



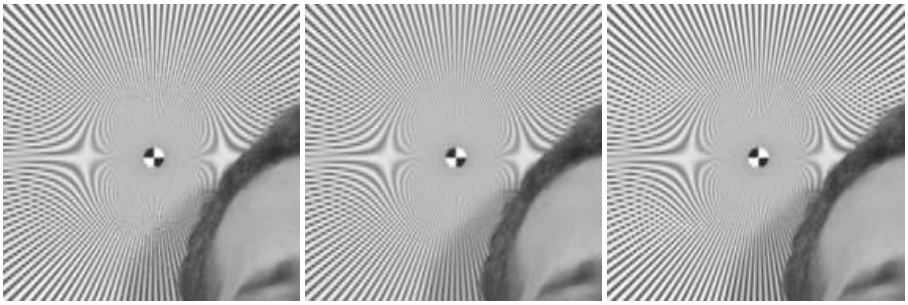
(a) (b) (c)

(d) (e) (f)

Figure 18: Comparison of the subjective performance of different deinterlacing methods with the proposed method: (a) Cropped original #4 Zahra image, (b) interlaced image, (c) MELA, (d) FFD, (e) FWAF_{τ} and (d) FWAF_{0}.



(a) (b) (c)



(d) (e) (f)

Figure 19: Comparison of the subjective performance of different deinterlacing methods with the proposed method: (a) Cropped original #24 Zahra image, (b) interlaced image, (c) MELA, (d) FFD, (e) FWAF_{τ} and (d) FWAF_{0}.

if the given image has different characteristics from the training set, the performance could become bad. On the other hand, the result images from the proposed FWAF_τ and FWAF_0 have the best quality since they are able to reduce artifacts compared with conventional methods.

From the objective and visual assessments of the proposed method, it can be concluded that our method preserves edge details and generates fewer artifacts than conventional methods.

4. Conclusion

This paper described an effective intra-field deinterlacing method which uses local variance estimator. With the obtained local variance values, homogeneity or heterogeneity feature is assigned to a region. In homogeneous areas, a conventional bilinear filter was utilized to interpolate the missing line. In heterogeneous areas, the fuzzy-based weighted average filter was applied to interpolate the missing line. It can be concluded that the proposed method can reflect the real local detail and intensity information by local variance estimator. Simulation results show that the proposed method can meaningfully enhance both the subjective and objective performances of reconstructed images.

References

[1] P. Brox, I. Baturone, S. Sanchez-Solano, and J. Gutierrez-Rios, "Edge-adaptive spatial video de-interlacing algorithms based on fuzzy logic," *IEEE Trans. Cons. Elect.*, vol. 60, no. 3, pp. 375–383, Aug. 2014.

[2] "Information Paper on HDTV Formats." Available: https://tech.ebu.ch/docs/techreports/tr005.pdf. Accessed 06 November 2014

[3] E. B. Bellars, and G. D. Haan, *De-interlacing: A Key Technology for Scan Rate Conversion*, Elsevier, Amsterdam, 2000.

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

[5] T. Doyle, "Interlaced to sequential conversion for EDTV application," in *Proc. 2nd Int. Working Processing of HDTV*, Feb. 1998, pp. 412-430.

[6] J. Kovacevic, R. J. Safranek, and E. M. Yeh, "Deinterlacing by successive approximations," *IEEE Trans. Image Process.*, vol. 6, no. 2, pp. 339–344, Feb. 1997.

[7] A. Skarabot, G. Ramponi, and D. Toffoli, "Image sequence processing for videowall visualization," in *Proc. SPIE Nonlinear Image Processing Xl*, vol. 3961, pp. 138–147, 2000.

[8] A. Skarabot, G. Ramponi, and L. Buriola, "FPGA architecture for a videowall image processor," in *Proc. SPIE Int. Symp. Electronic Imaging*, vol. 4304, pp. 75–84, 2001.

[9] G. Jeon, M. Anisetti, D. Kim, V. Bellandi, E. Damiani, and J. Jeong, "Fuzzy rough sets hybrid scheme for motion and scene complexity adaptive deinterlacing," *Image and Vision Computing*, vol. 27, no. 4, pp. 425–436, March 2009.

[10] G. Jeon, M. Anisetti, V. Bellandi, E. Damiani, and J. Jeong, "Designing of a type-2 fuzzy logic filter for improving edge-preserving restoration of interlaced-to-progressive conversion," *Information Sciences*, vol. 179, no. 13, pp. 2194–2207, June 2009.

[11] G. Jeon, M. Anisetti, J. Lee, V. Bellandi, E. Damiani, and J. Jeong, "Concept of linguistic variable-based fuzzy ensemble approach: application to interlaced HDTV sequences," *IEEE Trans. Fuzzy Systems*, vol. 17, no. 6, pp. 1245–1258, Dec. 2009.

[12] K. Kang, G. Jeon, and J. Jeong, "A single field interlaced to progressive format conversion," in *Proc. IASTED Signal and Image Processing (SIP)*, 2009, pp. 132–137.

[13] P.-Y. Chen and Y.-H. Lai, "A low-complexity interpolation method for deinterlacing," *IEICE Trans. Inf. & Syst.*, vol. E90-D, no. 2, pp. 606–608, Feb. 2007.

[14] W. Kim, S. Jin, and J. Jeong, "Novel intra deinterlacing algorithm using content adaptive interpolation," *IEEE Trans. Cons. Elect.*, vol. 53, no. 3, pp. 1036–1043, Aug. 2007.

[15] D.-H. Lee, "A new edge-based intra-field interpolation method for deinterlacing using locally adaptive-thresholded binary image," *IEEE Trans. Cons. Elect.*, vol. 54, no. 1, pp. 110–115, Feb. 2008.

[16] S. Yang, D. Kim, and J. Jeong, "Fine edge-preserving deinterlacing algorithm for progressive display," *IEEE Trans. Cons. Elect.*, vol. 55, no. 3, pp. 1654–1662, Aug. 2009.

[17] S.-J. Park, G. Jeon, and J. Jeong, "Covariance-based adaptive deinterlacing method using edge map," in *Proc. Image Processing Theory Tools and Applications (IPTA)*, 2010, pp. 166–171.

[18] G. Jeon and J.-K. Lee, "Filter design and its application for scanning format conversion," *SPIE Optical Engineering*, vol. 51, no. 4, pp. 040504, April 2012.

[19] J. Riquelme, S. Morillas, G. Peris-Fajarnes, D. Castro, "Fuzzy metrics application in video spatial deinterlacing," *Lecture Notes in Computer Science*, vol. 4578, 2007, pp 349–354.

[20] P. Brox, I. Baturone, S. Sanchez-Solano, "A motion and edge adaptive interlaced-to-progressive conversion using fuzzy logic-based systems," in *Proc. IPMU2008*, pp. 1175–1182.

[21] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[22] LC dataset: Available, http://www.gipsa-lab.grenoble-inp.fr/∼laurent.condat/imagebase.html

[23] "CIPR Still Images." Available:
http://www.cipr.rpi.edu/resource/stills/index.html.
Accessed 27 Mar 2013

[24] "Xiph.org Video Test Media [derf's collection]." Available: http://media.xiph.org/video/derf/. Accessed 27 Mar 2013

[25] McM dataset: Available,
http://www4.comp.polyu.edu.hk/∼cslzhang/CDM_Dataset.htm

[26] Zahra dataset: Available,
http://ivrl.epfl.ch/supplementary_material/cvpr11/index.html