Deinterlacing Algorithm Using Blackman-Harris Filter along the Piecewise Edge Direction

Joohyeok Kim and Jechang Jeong

***Abstract*—In this paper, we propose an improved deinterlacing algorithm which applies a filter along the edge directions. The existing deinterlacing algorithms do not take direction and rapidly changing pixels into account sufficiently and, in turn, it causes quality degradation. In order to improve image quality, we introduce piecewise-directional detection. It determines the direction between each line, which consists of nonlinear edge directions, in order to avoid the overshoot and undershoot. Also, we set the six filter coefficients using Blackman-Harris window to reconstruct a pixel to be interpolated. BH window mitigates the problem of truncated sinc functions that brings about Gibbs' phenomenon due to the limited input signal. Simulation results show that the proposed algorithm is superior to the existing algorithms.**

***Index Terms*—Deinterlacing, SINC interpolation, piecewise-direction.**

I. Introduction

We are witnessing a revolution like the revolution leading from black and white TV to color TV. This is the change from digital TV to High Definition TV (HDTV) in which viewers can watch high quality video with much better resolution. Many counties are already preparing for the next generation TV format including Ultra HDTV (UHDTV) that can display 2K, 4K picture as well as the highly anticipated 3DTV. As these UHD and 3D format provide realistic broadcasting, viewers can watch the contents as if they are in the real place.

Along with this revolutionary change, there is a huge surge in the demand for transmitting huge amounts of high resolution images. While HD resolution demands 746Mbps, 4K-UHD resolution needs 18 Gbps (3840x2160, YUV 4:4:4, 12bit, 60fps), which is about 24 times of HD, and this volume doubles even in 3DTV. Since the available bandwidth is limited, many TV systems such as National Television System Committee (NTSC) and Phase Alternating Line (PAL) employ the interlaced scan format that has vertically half the resolution. However, the interlaced image sequence is accompanied by visual artifacts such as flicker effect and jaggedness phenomenon. Because the visual artifacts are compounded by increasing resolution, we need to apply some effective deinterlacing methods.

Manuscript received October 20, 2012; revised November 21, 2012. This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency)" (NIPA-2012-C1090-1200-0010)

The authors are with the Department of Electronics and Computer Engineering, Hanyang University, Seoul, Korea (e-mail: kjh76363@gmail.com, jjeong@ece.hanyang.ac.kr)

Deinterlacing is a method converting interlaced scan format to progressive scan format, and this can be classified into two types according to the reference frame during interpolation: interfield deinterlacing method [10]-[11] utilizing the previous and/or next frame and intrafield deinterlacing method[1]-[8] deploying just the current frame. The interfield deinterlacing methods provide higher image quality than intrafield deinterlacing methods as they deploy not only spatial correlation but also temporal correlation; however, the computational complexity is high. On the other hand, the intrafield deinterlacing methods are suited for real-time application because they require much lower complexity. We target the intrafield deinterlacing method in this paper.

A great deal of intrafield deinterlacing algorithms have been proposed and there are three categories: linear interpolation, edge-based interpolation, and filter-based interpolation. Linear interpolation algorithms such as nearest neighboring (NN) and line average (LA) are very fast but have the problems like jaggedness stated in the previous paragraph because they do not consider any edge.

Edge-based line average (ELA) [1], efficient ELA (EELA) [2], and modified ELA (MELA) [3] are well-known edge-based interpolation algorithms. They detect the edge by using pixel differences of six pixels on the above and the below neighboring lines, and do interpolation along the direction. They are easy to implement and have low computational complexity, but interpolation in complex regions usually fails to yield any satisfying result due to the limited edge directions.

Filter-based interpolation methods utilize certain filters to detect edge direction or to reconstruct pixels. Fine directional deinterlacing (FDD) find the edge direction using modified Sobel filter [5]. Covariance-based adaptive deinterlacing (CAD) makes use of geometric duality concept and Wiener filtering theory for interpolation [7]. Hong et al. use fixed directional interpolation filter (FDIF) as the kernel for interpolation [8]. Filter-based interpolation algorithms outperform linear interpolations and edge-based interpolations but they cause considerable computational complexity.

The rest of the paper is organized as follows. We explain the piecewise-directional detection, the ideal sinc kernel, and BH sinc kernel in Section 2. The experimental results are presented to compare the proposed algorithm with conventional algorithms in Section 3. Finally, we conclude the paper in Section 4.

DOI: 10.7763/IJFCC.2013.V2.182

II. Proposed Algorithm

A. Blackman-Harris Windowed SINC Kernel

Since the information theory was published by Shannon, the SINC function has been widely accepted as the ideal interpolation [9]. A continuous signal $f(x, y)$ yields non-overlapped infinite repetitions of its continuous spectrum $F(u, v)$ in frequency domain, which means that any sampled signal can be reconstructed to the original continuous signal provided the *Ny* quist Sampling Theory (NST) is satisfied [12]. In other words, we can obtain the continuous signal $f(x, y)$ from the sampled signal $f_s(k, l)$ through multiplication with the rectangular function in the frequency domain, which is equal to convolution with SINC function in the spatial domain:

$$f(x) = \sum_k f_s(k) \cdot h(x-k), \tag{1}$$

$$h_{ideal}(x) = \frac{\sin \pi x}{\pi x} = \mathrm{sinc}(x). \tag{2}$$

As we know from the above equations, ideal kernel has one at x=0 and zero at all the integer positions. Also, it has positive values between zero and one, negative values between one and two, positive values from two to three, and so on. Fig. 1 shows the SINC function, which is the ideal kernel, and its magnitude of Fourier transform. The SINC kernel has no ripple in the stopband and retains value one along the passband.

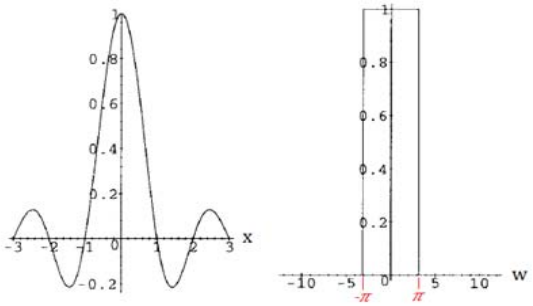

Fig. 1. SINC function and magnitude of Fourier transform.

Although the SINC function can reconstruct the original signal accurately, there is a challenge to its practical implementation. Sampled signal should be infinite in order to reconstruct the original continuous signal; however, only finite signals are available in image processing. In order to overcome this problem, windowed SINC function with a less severe window $w(x)$ than the rectangular function is used as kernel during spatial convolution. Hann-window, Kaiser-Bessel (KB) window, and Blackman-Harris (BH) window are some of the widely used windows, and we use BH window because its implementation is easy and it has less ripple in the stop band.

When filter length is six, BH window is represented as

$$w(n) = a_0 + a_1 \cos(\frac{2\pi}{N} n) + a_2 \cos(\frac{2\pi}{N} 2n), \tag{3}$$

where a_0 = 0.4266, a_1 = 0.4966, a_2 = 0.0768, and N is filter length.

BH windowed SINC kernel is formed using multiplication of SINC function and window function like:

$$h_{BH}(x) = \mathrm{sinc}(x) \cdot w(x), \quad 0 \le |x| \le 3. \tag{4}$$

Convolution of BH kernel with sampled pixel values from –N to N as shown in the equation (5) constructs an interpolated pixel at x position. We use 6tap filter (N=3) using this BH window, and its coefficients are [2, -11, 73, 73, -11, 2]/128.

$$f(x) = \sum_{k=-3}^{3} f(k) \cdot h(x-k). \tag{5}$$

B. Determination of Piece-Wise directions

Most of the linear interpolation algorithms and edge-based interpolation algorithms mentioned in Section I use pixels from the two lines of the above and the below neighboring lines. For this reason, they could not efficiently calculate the correlation of the image. As many filter-based interpolations just apply interpolation kernel based on uni-direction, they could not avoid overshoot or undershoot caused by using the pixel with little correlation. This usually occurs in the case when the neighboring pixels to be used have drastic difference like as Fig. 2. If they make a pixel located in 0.5 position (dashed red line) using the neighboring six pixels including the pixel at x=3, it will produce lower value than original value. It is because they do not consider the edge being between x=2 and x=3. In this case, it provides better result to use just four pixels at x=-1, 0, 1, 2 or to use five pixels instead of six pixels. To circumvent this over/undershoot, we interpolate the pixel using the pixels along the edge obtained by detecting the piecewise-directions.

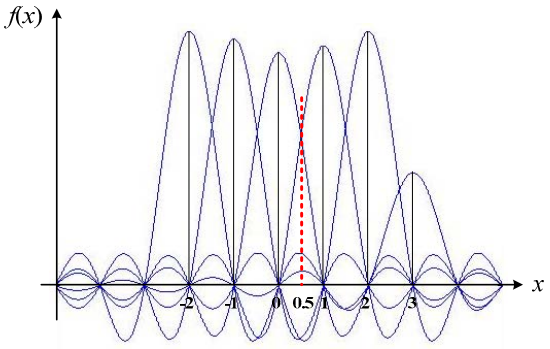

Fig. 2. Signal with drastic pixel difference.

$$\begin{aligned} D_S &= \big(|U1(-1) - D1(-1)| + |U1(0) - D1(0)| \\ &\qquad + |U1(1) - D1(1)|\big) \times \tfrac{2}{3} \\ D_L &= \big(|U1(-1) - D1(0)| + |U1(0) - D1(1)|\big) \\ D_R &= \big(|U1(0) - D1(-1)| + |U1(1) - D1(0)|\big) \end{aligned} \tag{6}$$

Determination of the piecewise-direction can be described as follows. We first find the direction in the U1-D1 region at the first stage, which is illustrated in Fig. 3. The distortions of straight line, left inclined line, and right inclined line (D_S, D_L, D_R in the equation (6)) are used to determine the direction.

If D_S of the first stage is less than predefined Th1, the to-be-interpolated pixel is assumed to be in the flat region or in the vertical direction, and we interpolate the pixel just by using the line average without more piecewise-direction detections.

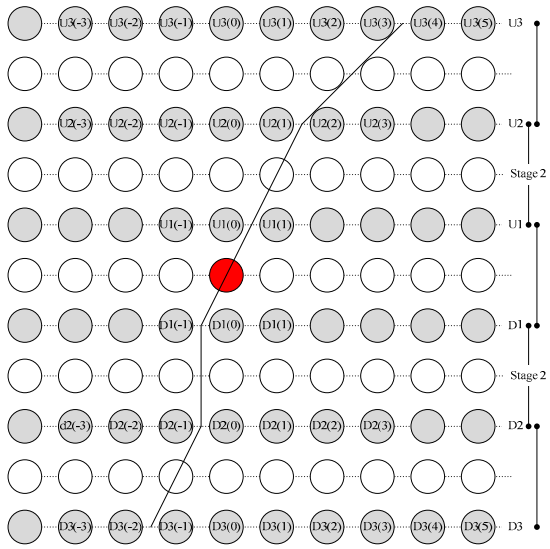

Fig. 3. Example of the piecewise-direction detection.

Fig. 3 shows the example of the piecewise-direction detection. D_R is the minimum among the three directions at the first stage in this figure and we assume that U1-D1 region has a slope of about sixty three degrees. Considering that the direction at the first stage is about 63°, we use the left two pixels (U2(0) and U2(1)) and the right two pixels (U2(2) and U2(3)) of the slope in order to determine the direction at the stage 2 (U1-U2 piece). This is represented in equation (7).

$$\begin{aligned} D_S &= |U2(1)-U1(0)|+|U2(2)-U1(1)| \\ D_L &= |U2(0)-U1(0)|+|U2(1)-U1(1)| \\ D_R &= |U2(2)-U1(0)|+|U2(3)-U1(1)| \end{aligned} \tag{7}$$

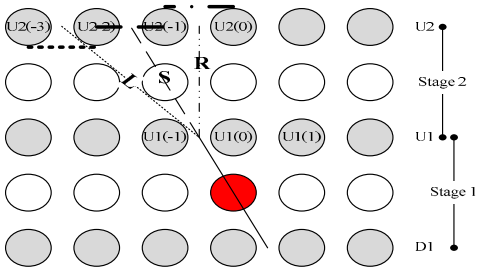

Fig. 4. Slope according to the previous direction.

The way to deduce the slope is as follows. We set the straight line from the previous direction (x/y=2, -63°) as S, the line shifted to left by one as L, and the right shifted line as R as shown in Fig. 4. According to this, S line uses two pixels, U2 (-2) and U2(-1), L line uses U2(-3) and U2(-2), and R line utilizes U2(-1) and U2(0) in the U1-U2 piece, respectively.

The same process is applied to the remaining pieces. The horizontal relative location of the current slope to the location of the slope at the previous stage is represented by equation (8). Because the stage 1 has the left direction in the Figure 4, the position of the slope on the U1 line is -1/2 (= 1/2×-1). If the stage 2 has the right direction, the position of the slope on the U2 line is -1/2 (= -1/2 -1 +1).

$$\Delta x_{stage} = \begin{cases} \frac{1}{2}d_1, & \text{in stage 1} \\ \Delta x_1 + d_1 + d_2, & \text{in stage 2} \\ \Delta x_2 + d_1 + d_2 + d_3, & \text{in stage 3} \end{cases} \tag{8}$$

where d_k is as follows:

$$d_k = \begin{cases} -1, & \text{left} \\ 0, & \text{straight} \\ 1, & \text{right} \end{cases}. \tag{9}$$

Note that we use $-d_k$ when we calculate the relative position in the below pieces. This is because the horizontal location of slope in the below regions is decreased when the edge is inclined to right. The equation (9) can be generalized like equation (10), and this piecewise-direction detection can be applied to larger filter length without any change.

$$\Delta x_{stage} = \begin{cases} \frac{1}{2}d_1, & \text{in stage 1} \\ \Delta x_{stage-1} + \sum_{k=1}^{stage} d_k, & \text{in stage=2,3,...} \end{cases} \tag{10}$$

After determining these piecewise-directions of five regions, U5-U3, U3-U1, U1-D1, D1-D3, D3-D5, we interpolate a pixel along those directions.

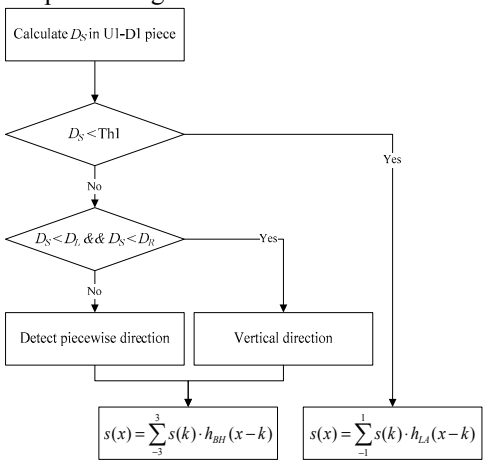

Fig. 5. The flowchart of the proposed algorithm.

The flowchart of the proposed algorithm is shown in Fig. 5. During the determination of piece-wise direction, the determination is terminated and we use just the valid pixels regardless of how many pixels are valid if the D value at the current stage is larger than the D value of the previous stage, which implies that the values in this line has changed drastically.

III. Experimental Results

We experimented the proposed algorithm and many conventional algorithms mentioned in Section 1 for comparison: LA, ELA, EELA, DOI, MELA, and FDD. We include objective results measured by peak signal-to-noise

ratio (PSNR):

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{255^2}{\mathrm{MSE}}\right), \quad (11)$$

where MSE denotes the mean squared error. The nine still images used in this experiment are airplane, baboon, Barbara, boat, lena, mobcal, peppers, temple, and toy. We interpolate these images from 512x256 size to 512x512 size. Table I shows the PSNR results. As shown in Table I, the proposed algorithm provides the best averaged PSNR results. Compared to FDD that has good PSNR performance, the proposed algorithm showed a positive gain of 0.29dB. In the image Baboon, some algorithms such as LA and FDD that just use neighboring pixels give better results. This is because Baboon image has very high frequency like salt and pepper noise, and its pixel values change drastically and very frequently.

TABLE I: PSNR RESULTS

	LA	ELA	EELA	DOI	MELA	FDD	Prop.
Airplane	34.23	33.1	33.92	34.21	34.10	34.22	34.84
Baboon	23.93	23.37	23.72	23.94	23.89	23.94	23.64
Boat	35.39	32.46	34.57	35.38	35.27	35.43	36.12
Lena	37.68	36.03	37.35	37.68	37.93	38.03	38.29
Mobcal	32.08	31.14	31.94	32.08	32.08	32.11	32.37
Peppers	33.79	34.10	34.32	33.80	34.13	33.95	33.95
Temple	33.37	32.30	33.20	33.36	33.29	33.28	33.60
Toys	33.34	32.59	33.17	33.34	33.33	33.36	33.80
Average	32.97	31.88	32.77	32.97	33.00	33.04	33.33

Table II shows the results of CPU time. We can see that the proposed algorithm consumes more time than the conventional algorithms based on LA, but provides better results than other algorithms like FDD.

TABLE II: CPU TIME RESULTS

	LA	ELA	EELA	DOI	MELA	FDD	Prop.
Airplane	0.031	0.031	0.031	0.312	0.296	2.855	1.740
Baboon	0.015	0.031	0.031	0.905	0.296	3.869	3.050
Boat	0.031	0.031	0.031	0.312	0.296	3.198	2.040
Lena	0.031	0.031	0.047	0.187	0.312	2.745	1.553
Mobcal	0.015	0.047	0.047	0.328	0.328	2.993	1.875
Peppers	0.015	0.031	0.047	0.203	0.312	2.745	1.521
Temple	0.031	0.031	0.047	0.343	0.281	3.120	1.946
Toys	0.031	0.031	0.047	0.296	0.328	2.964	1.511
Average	0.025	0.033	0.041	0.036	0.306	3.061	1.904

IV. CONCLUSION

In this paper, we proposed an deinterlacing algorithm that determines the edge direction in the region between a line and its neighboring lines, we call it a piecewise-direction, and interpolate a pixel using BH filter. By using piecewise-direction detection, we can exclude pixels that might have bad influence to the to-be-interpolated pixel due to its drastic pixel value change. Because the sinc function requires unlimited input signal, it is difficult to apply to the interpolation; hence, we use BH sinc filter. BH filter is formulated by multiplying a smoothing window to sinc filter in the frequency domain in order to mitigate the Gibbs' phenomenon. Simulation results showed that the proposed algorithm is superior to the existing algorithms.

ACKNOWLEDGMENT

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency)" (NIPA-2012-C1090-1200-0010)

REFERENCES

[1] 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, Jun. 1996.
[2] T. Chen, H. R. Wu, and Z. H. Yu, "Efficient deinterlacing algorithm using edge-based line average interpolation," *Opt. Eng*., vol. 39, no. 8, pp. 2101– 2105, Aug. 2000.
[3] 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.
[4] H. Yoo and J. Jeong, "Direction-oriented interpolation and its application to de-interlacing," *IEEE Trans. Cons. Elect.*, vol. 48, no. 4, pp. 954–962, Nov. 2002.
[5] S. Jin, W. Kim, and J. Jeong, "Fine directional de-interlacing algorithm using modified Sobel operation," *IEEE Trans. Cons. Elect*., vol. 54, no. 2, pp. 857–862, May. 2008.
[6] A. Fuldseth, G. Bjontegaard, D. Rusanovskyy, K. Ugur, and J. Lainema, "Low complexity directional interpolation filter," *ITU-T SG16/Q.6* Doc. VCEG-AI12, pp. 16–18.
[7] S. J. Park, G. Jeon, and J. Jeong, "Computation-aware algorithm selection approach for interlaced-to-progressive conversion," *SPIE Opt. Eng*., vol. 49, no. 5, pp. 057005:1-9. May 2010.
[8] S. Hong, S. Park, and J. Jeong, "Deinterlacing algorithm using fixed directional interpolation filter and adaptive distance weighting scheme," *Opt. Eng*., vol. 50, no. 6, Jun. 2011.
[9] T. M. Lehmann, C. Gonner, and K. Spitzer, "Survey: Interpolation methods in medical image processing," *IEEE Trans. Medi. Imag*., vol. 18, no. 11, pp. 1049-1075, Nov. 1999.
[10] Y. Y. Jung, B. T. Choi, Y. J. Park, and S. J. Ko, "An effective deinterlacing technique using motion compensated interpolation," *IEEE Trans. Cons. Elect*., vol. 46, no. 3, pp. 460-466, Aug. 2000.
[11] K. Sugiyama and H.Nakamura, "A method of deinterlacing with motion compensated interpolation," *IEEE Trans. Cons. Elect*., vol. 45, no. 3, 611-616, Aug. 1999.
[12] F. J. Harris, "On the use of windows for harmonic analysis with the discrete fourier transform," in *Proc. of the IEEE*, vol. 66, no. 1, pp. 51-83, Jan. 1978.
[13] F. Stenger. *Numerical methods based on SINC and analytic functions*. U.S: Spinger, Jun. 1993.

Joohyeok Kim received a BS degree in information communication from Hanyang University, Korea, in 2008. Currently, he is a PhD candidate in the Department of Electronics and Computer Engineering, Hanyang University. His research interests include image processing, video coding standards such as H.264/AVC, HEVC and 3DV, and image enhancement, as well as image resizing algorithms.

Jechang Jeong received a BS degree in electronic engineering from Seoul National University, Korea, in 1980, an MS degree in electrical engineering from the Korea Advanced Institute of Science and Technology in 1982, and a PhD degree in electrical engineering from the University of Michigan, Ann Arbor, in 1990. Since 1995, he has conducted research at Hanyang University, Seoul, Korea. His research interests include digital signal processing, digital communication, and image/audio compression for HDTV and multimedia applications. He has published over 30 technical papers. He received the "Scientist of the Month" award in 1998 from the Ministry of Science and Technology of Korea. He was also honored with a government commendation in 1998 from the Ministry of Information and Communication of Korea. He was the recipient of the IEEE Chester Sall Award in 2007 and the ETRI Journal Paper Award in 2008.