Perception-inspired spatio-temporal video deinterlacing

Ragav Venkatesan[1], Christine Zwart[2], David Frakes[2,3] *Member, IEEE*, Baoxin Li[1] *Senior Member, IEEE*

[1]School of Computing Informatics and Decision Systems Engineering, Arizona State University, Tempe, AZ, USA
[2]School of Biological and Health Systems Engineering, Arizona State University, Tempe, AZ, USA
[3]School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA

I. Introduction

DEINTERLACING is the process of converting an interlaced video format to a progressive video format. Interlaced video formats can be very useful when bandwidth is limited and are also well-suited for scanning display systems. Interlaced videos are scanned in such a way that in any given frame with N rows, only $N/2$ alternate rows are present. The remaining rows are scanned in the next frame, and when the frames are displayed quickly enough, humans are unable to detect the missing lines (since the human eye doesn't update quickly enough). Interlaced videos are generally preferred in video broadcast and transmission systems. Interlaced videos are also preferred in high motion videos where vertical frequency is compromised to get a higher frame rate.

Video interlacing motivates many tasks pertaining to international TV broadcasting, such as format conversion. Moreover, many modern display systems work on progressive video streams and thus require a deinterlacer. Poor deinterlacing can be observed today in a wide range of consumer products. Figure 1 shows such a product from a recent YouTube video. Even though deinterlacing is a traditional topic in video processing and numerous approaches have been taken to solve the problem, there is a renewed interest due to recent developments in high speed and dedicated video processing hardware in display systems.

Bellers and Haan defined deinterlacing formally as:

$$\hat{F}_n(i,j) = \begin{cases} F_n(i,j), & j \bmod 2 = n \bmod 2 \\ F_n^I(i,j), & \text{otherwise,} \end{cases} \quad (1)$$

where F_n is the original interlaced video, F_n^I is the interpolated video, $\hat{F}_n$ is the deinterlaced video, n is the frame index, and (i,j) are the spatial pixel indices. It is the interpolator estimating $F_n^I(i,j)$ that the deinterlacer's quality depends on.

Based on the type of interpolator that estimates $F_n^I(i,j)$, deinterlacers can be classified as spatial, temporal, or a combination of both. Spatial interpolators interpolate within a given frame and are usually preferred when there is a high degree of motion in the video. In such cases, the content of the video changes too quickly for temporal interpolators to perform well. Temporal interpolators work exclusively across frames and work well when there is little motion. Most modern deinterlacers employ method switching algorithms that use different estimates or combinations of different estimates from different interpolators for particular regions of video. Motion in the video is usually the preferred basis for method switching; in a region of video with high motion a near-spatial interpolator is preferred. In this paper, we propose not a motion-based approach, but rather a perception-inspired approach to such interpolator selection.

Fig. 1. Example of poor deinterlacing from a high-definition YouTube video.

The regions of video that are perceptually salient are those that the human eye fixates upon and are thus effectively updated more often by the human visual system than those regions that are not perceptually salient. Good cinematographers ensure that the region with most activity is always salient [1]. With this understanding, it follows that the salient regions of the video, those that need to be updated more often, are better off interpolated using data from as small a temporal window as possible (preferably from within the same frame). While a purely background pixel that doesn't change across two frames can be fully temporally-averaged well, the non-salient regions can more effectively be interpolated in a spatio-temporal manner. The basis for the proposed algorithms follows this argument.

Spectral residue in the context of perception is quite well studied [2]. The quaternion Fourier implementation of spectral residue was studied first by Zhang et al. [3]. In this paper, we use a similar saliency map and spectral residue in weighting to linearly combine spatial and temporal interpolator contributions. The spatial and temporal interpolators that we use are $1D$ control grid interpolator ($1DCGI$), and $2D$ control grid interpolator ($2DCGI$), respectively [4] [5].While $1DCGI$ is an intra-frame optical flow based interpolator that works like an edge-directed interpolator, $2DCGI$ is a more traditional

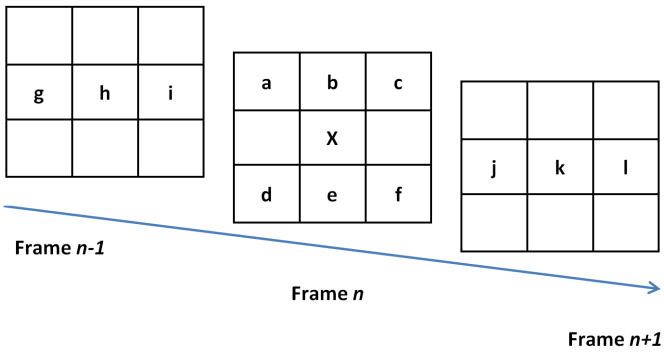

Fig. 2. Neighborhoods for STELA and ELA.

optical flow-based temporal interpolator. While neither one of these methods alone is best for deinterlacing, a combination of the two yields perceptually beneficial results.

The rest of the paper is organized as follows: section II covers related works, section III explains the proposed approaches, section IV describes the experiments, section V documents the results, and section VI provides concluding remarks.

II. Related works

A straight forward temporal deinterlacer takes the form:

$$\hat{F}_n^{LA}(i,j) = \begin{cases} F_n(i,j), & j \bmod 2 = n \bmod 2 \\ \frac{F_{n-1}(i,j)+F_{n+1}(i,j)}{2}, & \text{otherwise,} \end{cases} \tag{2}$$

This method is called the Temporal line average (LA), simply LA, or the *bob* algorithm. The algorithm performs well when there is very little motion. Many modern method switching algorithms still incorporate LA as one of the methods when the difference across two frames is lower than a threshold.

A fully spatial non-linear interpolator that works within a small window is the Extended LA or Edge-based LA(ELA) [6] [7]. Figure 2 shows the window of operation of ELA. While interpolating for the point X, three directional differences are estimated as $C1 = |a-f|, C2 = |b-e|$, and $C3 = |c-d|$ where a, b, c, d, e, and f are defined as in Figure 2. The minimum difference among $C1$, $C2$, and $C3$ is chosen. The interpolated value for X is then the average of the two points that corresponded to the minimum difference.

Many edge-based interpolators similar to ELA have also been proposed. One efficient ELA implementation (EELA) uses directional spatial correlation instead of angular edge directions [8]. The low complexity interpolation method for deinterlacing (LCID) uses four directions rather than the three used in ELA [9]. Instead of estimating edge directions using differences, LCID uses the edges from a sobel filtered image and interpolates along the detected edges [10].

Spatio-temporal edge-based median filtering (STELA) adds a temporal component to an intra-frame deinterlacer like ELA [11]. STELA is a two-pronged approach. It divides a video frame into low frequency and high frequency frames. In the low frequency frame, STELA works on a $3X3X3$ neighborhood as shown in Figure 2. It estimates six directional differences, unlike ELA that works with only three. The six directional differences are $C1 = |a-f|, C2 = |b-e|, C3 = |c-d|, C4 = |g-l|, C5 = |h-k|$, and $C6 = |i-j|$. The deinterlaced estimate for any point $X = Med\{A, b, e, h, k\}$, where A is the average value of the two points that yield the minimum directional change among $C1$ through $C6$ and Med is a median operator. Although A is the preferred value for X, the median filter is added as a backup in case there is noise in the video. Whenever there is noise in the video and that alters the decision to choose A, the median eliminates the noisy pixel and still provides an acceptable result. The high frequency frames are subject to line doubling or *weaving*. The line doubled version is added to the processed low frequency frames. STELA showed that spatio-temporal methods work better than purely spatial deinterlacers like ELA when the interlaced video contains both low-motion background regions and fast-changing foreground regions.

A computationally efficient spatio-temporal deinterlacer is the vertical temporal filter (VTF) [12]. VTF is a filtering algorithm and is defined as:

$$\hat{F}_n^{VTF}(i,j) = \begin{cases} F_n(i,j), & j \bmod 2 = n \bmod 2 \\ \sum_m \sum_k F_{n+m}(i, j+k) h_m(k), & \text{else,} \end{cases} \tag{3}$$

where Weston proposed the filter $h_m(k)$ to be:

$$h_m(k) = \begin{cases} \frac{1}{2}, \frac{1}{2} & (k = -1, 1 \ \& \ m = 0) \\ -\frac{1}{16}, \frac{1}{8}, -\frac{1}{16} & (k = -2, 0, 2 \ \& \ m = -1, 1). \end{cases} \tag{4}$$

VTF is not an adaptive algorithm like STELA or ELA but is still among the most popular deinterlacing algorithms because of its computational efficiency. Adaptations of the algorithm are seen in deinterlacing as late as 2013. Content adaptive VTF (CAVTF) and spatially registered VTF or SRVTF are two examples [13] [14].

CAVTF is a two-step algorithm where each pixel is classified into one of three classes by using a modified adaptive dynamic range encoding. Once each pixel is classified and provided sufficient temporal differences exist, an adaptive version of VTF is implemented wherein filter values depends on the neighborhood pixel values. SRVTF is a VTF algorithm applied not to the interlaced video but to spatially registered frames. A global motion estimation is performed to estimate motion vectors v_x and v_y as:

$$(v_x^*, v_y^*) = \underset{(v_x, v_y) \in MV}{\operatorname{argmin}} \sum |F_{n-1}(i,j) - F_{n+1}(i+v_x, j+v_y)|, \tag{5}$$

where the motion vectors don't span more than 8 pixels in either direction ($\{(v_x, v_y) | -8 \le v_x, v_y \le 8; v_x, v_y$ are even$\}$). After estimating the motion vectors, spatial registration is performed as:

$$F_{n-1}^{SR}(i,j) = F_{n-1}(i - v_x^*/2, j - v_y^*/2) \tag{6}$$

and

$$F_{n+1}^{SR}(i,j) = F_{n+1}(i + v_x^*/2, j + v_y^*/2). \tag{7}$$

Traditional VTF is performed on the spatially registered frames F_n^{SR} to get $\hat{F}_n^{SR}(i,j)$. A frame-difference-like technique is used as a reality check to make sure that the registered

frames do perform better than the original VTF. The frame differences are $d1$ and $d2$, which are defined as:

$$\begin{aligned} 3d_1 = &|F^{SR}_{n-1}(i,j-2) - F^{SR}_{n+1}(i,j-2)| \\ &+ |F^{SR}_{n-1}(i,j) - F^{SR}_{n+1}(i,j)| \\ &+ |F^{SR}_{n-1}(i,j+2) - F^{SR}_{n+1}(i,j+2)| \end{aligned} \quad (8)$$

and

$$\begin{aligned} 3d_2 = &|F_{n-1}(i,j-2) - F_{n+1}(i,j-2)| \\ &+ |F_{n-1}(i,j) - F_{n+1}(i,j)| \\ &+ |F_{n-1}(i,j+2) - F_{n+1}(i,j+2)|. \end{aligned} \quad (9)$$

Deinterlacing is performed as:

$$\hat{F}^{SR}_n(i,j) = \begin{cases} \sum_m \sum_k F^{SR}_{n+m}(i,j+k)h_m(k) & if(d_1 < d_2) \\ \hat{F}^{VTF}_n & else. \end{cases} \quad (10)$$

The reasoning behind registration is that compensation for motion yields more suitable pixel neighbors for VTF to work with. This along with a second level verification using the frame differences, which gives the option to revert back to the original VTF, makes the algorithm robust.

While VTF is a fixed range filter, a non-local means filter-based approach was proposed by Wang et al. that estimates a missing pixel using an adaptive weighted average of all pixels in a patch-matched neighborhood [15]. By choosing an optimal range for the patch matching algorithm, good performance is achieved without compromising efficiency. Hong et al. use a similar distance-based weighting scheme to weight their sinc based interpolator [16]. An example of a purely motion-based approach is deinterlacing using hierarchical motion analysis [17]. This method uses motion analysis (in four-stages), pixel estimation, and pixel correction procedure to generate a likely pixel estimate. Although this method performs well, it is computationally expensive.

III. Proposed Algorithms

The proposed algorithm is a method switching approach that chooses either a temporal average or a linearly weighted combination of spatially and temporally interpolated estimates. The spatially interpolated estimate is generated with $1DCGI$ and the temporally interpolated estimate with $2DCGI$ [4] [5]. The choice is based on a threshold frame difference and the linear weights are the normalized spectral residues. The core of the proposed method is the use of spectral residue to make a choice between the spatial and temporal interpolators. The link between spectral residue and human perception is studied in [2]. Spectral residues for color images are estimated using the quaternion Fourier transform approach in [3]. The quaternion Fourier transform of an image is studied in [18]. Any color image can be represented using quaternions of the form:

$$q^n = Ch^n_1 + Ch^n_2\mu_1 + Ch^n_3\mu_2 + Ch^n_4\mu_3, \quad (11)$$

Fig. 3. Mother video (left) and the detected saliency (right) after thresholding by B=4% of the bit depth.

where μ_k for $k = 1, 2, 3$ satisfies $\mu_k^2 = -1, \mu_1 \perp \mu_2, \mu_2 \perp \mu_3$, and $\mu_1 \perp \mu_3$. The three color channels of an image can be allocated to Ch_2, Ch_3, and Ch_4, respectively, while Ch_1 is set to zero.

The quaternion Fourier transform (QFT) of an image is:

$$Q^n(u,v) = \frac{1}{\sqrt{WH}} \sum_{j=0}^{W-1} \sum_{i=0}^{H-1} e^{\mu_1 2\pi(\frac{jv}{W} + \frac{iu}{H})} q^n(i,j) \quad (12)$$

and its inverse is:

$$q^n(i,j) = \frac{1}{\sqrt{WH}} \sum_{v=0}^{W-1} \sum_{u=0}^{H-1} e^{\mu_1 2\pi(\frac{jv}{W} + \frac{iu}{H})} Q^n(u,v), \quad (13)$$

where $q[i,j]$ are samples in the spatial domain, $Q[u,v]$ are samples in frequency domain, and W and H are the width and height of the image in pixels, respectively. The phase spectrum of an image can be extracted by $Q^{phase} = \frac{Q}{||Q||}$. An approximation to spectral residue can be obtained by Gaussian smoothing the inverse QFT, q^{phase}. The L_1 norm of such a smoothed phase is also a measure of the visual saliency of the image [3]. Since we use the the spectral residue for weighting between spatial and temporal interpolators, we normalize the spectral residue as:

$$S_n(i,j) = \frac{||g * q^{phase}_n(i,j)||_1}{\max(||g * q^{phase}_n(i,j)||_1)}. \quad (14)$$

An example of the resulting saliency map is shown in Figure 3. Unlike SRVTF that uses motion as a region classifier, we use the spectral residue. Two kinds of deinterlacers are thus formulated: a hard decision deinterlacer (HDD) that uses the threshold by B saliency map and a soft decision deinterlacer (SDD) that uses the normalized spectral residue. These proposed approaches are to be elaborated in Section III-C. Since these approaches are built upon two interpolators that operate in 1D and 2D respectively, we first briefly describe them in Section III-A and III-B.

A. 1D Control Grid Interpolator

The 1D control grid interpolator ($1DCGI$) is based on the brightness constraint, similar to optical flow [4]. This assumption dictates that the intensity associated with any given location in the source data set is preserved and located somewhere in the destination data set. The vector connecting the source and destination defines the local transformation

that relates the two sets. Interpolation is performed by placing distance-weighted averages of the source and destination intensities along the "displacement" vector and then using convolution gridding to assign intensities at the unknown pixel locations.

The term displacement is used to describe the offset between the destination location and the nearest neighbor to the source location in the destination set. For example, defining the horizontal offset between adjacent lines as α, we write:

$$I(i,j) = I(i+\alpha, j+1). \tag{15}$$

The Taylor series expansion is used to represent the brightness constraint in terms of the displacement as a scalar:

$$I(i,j) \approx I(i,j) + \frac{\partial I(i,j)}{\partial x}\alpha + \frac{\partial I(i,j)}{\partial y}(1), \tag{16}$$

where x and y are taken as the horizontal and vertical axes corresponding to the indexing variables i and j respectively. Direct approaches to solving Equation 16 are sensitive to noise. Rather than address the error associated with each pixel displacement individually, smoothness is ensured by defining the displacements at regularly spaced control points, or nodes, and generating the intermediate displacements with linear interpolation. The full details of the control grid approach are covered in previous publications [4], [19].

In the deinterlacing application, matches are made between the data containing rows as:

$$I(i,j) = I(i+2\alpha_+, j+2), \tag{17}$$

or

$$I(i,j) = I(i-2\alpha_-, j-2), \tag{18}$$

where rows $j-2$, j and $j+2$ are known and α_+ and α_- define the horizontal displacements in each independent equation. For each case, α is used to directionally interpolate new values at each missing pixel. The two candidate values are equally weighted in constructing the final, complete frame.

The brightness constraint based $1DCGI$ is in practice a straight forward line-to-line edge directed interpolator that is comparable in style to ELA. In both cases, interpolation is carried out between pixels in data-filled rows selected to have minimal intensity differences. In contrast to 1DCGI, the candidate pixels for ELA are limited to a discrete subset (displacements are required to be integers) significantly reducing the angular resolution of the interpolated edge direction.

B. 2D Control Grid Interpolation

$2DCGI$ is defined by the following embodiment of the $2D$ optical flow equation:

$$I[i,j,k] = I(i + d_1[i,j,k], j + d_2[i,j,k], k+\delta k). \tag{19}$$

The image is divided into grids and the horizontal and vertical pixel displacements within each block are modelled as:

$$d_1(i,j) = \sum_{l=1}^{p} \alpha_l \Theta_l(i,j) \tag{20}$$

and

$$d_2(i,j) = \sum_{l=1}^{p} \beta_l \Phi_l(i,j), \tag{21}$$

where $\Theta_l(i,j)$ and $\Phi_l(i,j)$ are independent basis functions that model the displacement field, and α and β are components of the velocity vector at each grid corner. When the control points (block corners) are shared across grids, the result is a piece-wise smooth and globally continuous motion model.

Analogous to splitting $1DCGI$ into top-down and bottom-up approaches, two displacement fields are constructed with $2DCGI$, one from frame k to $k+\delta k$ and another from frame $k+\delta k$ to k. This leads to two reconstructed images that are combined in a spatially weighted sum to create the final interpolated image.

C. Proposed Switching Schemes

The HDD can be formulated by using $1DCGI$ for salient regions in the video and VTF for other regions of video provided there is sufficient difference in pixel values across frames. Such an approach was first discussed by Venkatesan et al. and is described by the following equation [20]:

$$\hat{F}_n^{HDD}(i,j) = \begin{cases} F_n(i,j), & j \bmod 2 = n \bmod 2 \\ \frac{F_{n-1}(i,j)+F_{n+1}(i,j)}{2}, & D_n(i,j) < T \\ \sum_m \sum_k F_{n+m}(i,j+k)h_m(k), & S_n(i,j) < B; \\ & D_n(i,j) \geq T \\ 1D_n(i,j), & \text{else,} \end{cases} \tag{22}$$

where $h_m(k)$ is Weston's VTF, $D_n(i,j)$ is frame difference, $S_n(i,j)$ is the spectral residue, and $1D_n(i,j)$ is the $1DCGI$ estimate.

The SDD can be obtained by linearly weighting $1DCGI$ and $2DCGI$ estimates. The spatially-salient regions in a video are those particular regions that the human eye localizes first and that therefore demand the sharpness of a spatial interpolator. The non-salient regions take time for the human eye to register and are therefore handled sufficiently well by a more smoothing temporal interpolator. Thus, the linear choice is made as $1D_n(i,j)S_n(i,j) + 2D_n(i,j)(1-S_n(i,j))$. Whenever the frame difference $D_n(i,j)$ across two frames is lower than the threshold T, for example two intensity units, a frame average is performed. The SDD is formulated as:

$$\hat{F}_n^{SDD}(i,j) = \begin{cases} F_n(i,j), & j \bmod 2 = \\ & n \bmod 2 \\ \frac{F_{n-1}(i,j)+F_{n+1}(i,j)}{2}, & D_n(i,j) < T \\ 1D_n(i,j)S_n(i,j) & \\ +2D_n(i,j)(1-S_n(i,j)), & D_n(i,j) \geq T, \end{cases} \tag{23}$$

where $1D_n(i,j)$ is the 1DCGI estimate of the n^{th} frame, $2D_n(i,j)$ is the 2DCGI estimate. This method avoids the ambiguity of spatio-temporal interpolators like VTF and instead uses a straightforward combination of a purely spatial interpolator and a purely temporal interpolator that is based

TABLE I
TABLE OF PSNR. ALL THE METHODS IN THIS TABLE WERE IMPLEMENTED BY THE AUTHORS. CARE WAS TAKEN TO ENSURE THAT THE METHODS WERE IMPLEMENTED TO THE FINEST DETAIL PROVIDED IN THE RESPECTIVE PAPERS.

Video	STELA	VTF	SRVTF	HDD	SDD
Akiyo	41.237	41.117	41.364	47.301	49.212
Bowing	37.013	40.962	40.726	46.122	42.659
Bridge Far	38.788	33.689	34.308	42.423	37.833
Container	35.479	31.055	32.821	46.417	46.394
Deadline	35.662	33.152	33.009	42.814	39.154
Foreman	31.467	32.202	33.802	36.957	37.183
Galleon	31.609	27.058	27.163	42.048	41.758
Hall Monitor	36.942	32.023	35.027	41.892	38.578
Mother	42.599	38.058	41.635	45.635	44.813
News	36.855	39.088	38.045	44.597	41.539
Students	37.086	33.436	33.954	45.173	42.887
Paris	30.943	28.934	29.010	33.799	35.344
Sign Irene	36.181	36.401	37.413	40.108	38.381

TABLE II
TABLE OF VSNR VALUES. ALL THE METHODS IN THIS TABLE WERE IMPLEMENTED BY THE AUTHORS. CARE WAS TAKEN TO ENSURE THAT THE METHODS WERE IMPLEMENTED TO THE LAST DETAIL PROVIDED IN THE RESPECTIVE PAPERS.

Videos	VTF	SRVTF	HDD	SDD
Akiyo	43.21	43.15	46.81	47.21
Bowing	36.97	36.87	47.27	47.34
Bridge Far	31.77	30.96	41.80	41.81
Container	30.12	29.94	44.16	44.21
Foreman	30.93	30.37	37.70	37.78
Galleon	26.39	26.37	45.44	45.46
Hall	32.52	31.95	43.54	43.51
News	41.02	40.85	47.36	47.69
Paris	26.93	26.94	40.16	39.21
Sign Irene	33.01	32.85	35.19	35.98
Students	28.92	29.02	42.40	42.59

on the spectral residue. The more salient the region is, the higher the spectral residue and the more weight the spatial estimate gets, and vice-versa. The result is a smoother and higher quality deinterlaced video.

IV. EXPERIMENTS

The proposed algorithms were all implemented in MATLAB along with SRVTF. The test video set comprised of 13 commonly used CIF videos from the trace video library [21]. These videos were manually and deliberately interlaced, and then deinterlaced using different algorithms. It is reasonable to conclude from deinterlacing literature that when interlacing a video manually, videos can be considered to be interlaced in either of two ways:

1) Fields $n-1$ and n are split from the same frame. A deinterlaced frame is to be reconstructed into full resolution from the two interlaced fields. Two fields map to one deinterlaced frame and no data is lost while interlacing.
2) Fields $n-1$ and n are down-sampled from two unique frames (frames $n-1$ and n, respectively). One unique de-interlaced frame is to be reconstructed for every field. One field maps to only one frame and half the data is simply thrown away while interlacing.

We interlaced the videos by using the second method. This enabled us to maintain the number of frames, facilitating the use of reference-based computational metrics for evaluation. We calculated the following computational metrics for each of the methods:

1) Peak signal-to-noise ratio (PSNR).
2) Visual signal-to-noise ratio (VSNR) [22].

V. RESULTS

Table I compares the PSNRs of different deinterlacing algorithms and shows that HDD and SDD outperform the other algorithms. SDD performs particularly well on videos containing relatively clearly defined saliency, which also agrees with the saliency model we used. Although a study of various computational saliency models and their effects on region-selection for various deinterlacing methods is outside the scope of this article, it is noteworthy that with more accurate saliency models the visual quality of the proposed methods should be better.

HDD is a hard-choice algorithm that uses one or another estimate and has a higher PSNR on average. However, the PSNR performance of HDD doesn't necessarily prove its performance in terms of visual quality. VSNR is used to compare the methods for visual quality [22] [23]. Table II shows the results of VSNR. Based on these metrics, SDD keeps up with and often outperforms HDD. We achieve this result through linear weighting, which provides smoother deinterlacing than hard choices.

Figure 4 shows the deinterlaced output for one frame of some of the test videos. In the students video, while regions like the edge of the table were deinterlaced smoothly by the proposed methods, the other methods produce jagged edges. In the same video, the hand (which is a non-salient region) was affected by motion artifacts even using the proposed methods. This is because the hand, being a non-salient region, was interpolated with more weight for the temporal than for the spatial interpolator. In the foreman video, the diagonal edges in the wall and the Siemens logo which are non-salient regions were more smoothly deinterlaced with the proposed methods than with SRVTF.

VI. CONCLUSIONS

In this paper, we propose a perception-inspired saliency-based approach to spatio-temporal deinterlacing. We use spectral residue in weighting the established temporal and spatial interpolators $2DCGI$ and $1DCGI$. The proposed method was compared against the state-of-the-art using a traditional computational metric (PSNR) and a visual quality metric (VSNR). All results showed that the proposed method outperforms the state-of-the-art.

Fig. 4. Video screenshots corresponding to different algorithms. From left to right are original, VTF, SRVTF, HDD, and SDD. From top to bottom are original and deinterlaced versions of frame 2 from foreman and students videos. The performance of the proposed approaches can be best appreciated on the edges on the wall and the siemens logo in the foreman video(top), and on the edges on the table in the students video(bottom).

REFERENCES

[1] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 11, pp. 1254–1259, 1998.

[2] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 2007, pp. 1–8.

[3] C. Guo, Q. Ma, and L. Zhang, "Spatio-temporal saliency detection using phase spectrum of quaternion fourier transform," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[4] C. M. Zwart and D. H. Frakes, "One-dimensional control grid interpolation-based demosaicing and color image interpolation," in *Proc. SPIE*, vol. 8296, 2012, p. 82960E.

[5] D. H. Frakes, L. P. Dasi, K. Pekkan, H. D. Kitajima, K. Sundareswaran, A. P. Yoganathan, and M. J. Smith, "A new method for registration-based medical image interpolation," *Medical Imaging, IEEE Transactions on*, vol. 27, no. 3, pp. 370–377, 2008.

[6] T. Doyle, "Interlaced to sequential conversion for edtv applications," in *Proc. 2nd int. workshop signal processing of HDTV*, 1990, pp. 412–430.

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

[8] T. Chen, H. R. Wu, and Z. H. Yu, "Efficient deinterlacing algorithm using edge-based line average interpolation," *Optical Engineering*, vol. 39, no. 8, pp. 2101–2105, 2000.

[9] C. Pei-Yin and L. Yao-Hsien, "A low-complexity interpolation method for deinterlacing," *IEICE transactions on information and systems*, vol. 90, no. 2, pp. 606–608, 2007.

[10] H. Yoo and J. Jeong, "Direction-oriented interpolation and its application to de-interlacing," *Consumer Electronics, IEEE Transactions on*, vol. 48, no. 4, pp. 954–962, 2002.

[11] H.-S. Oh, Y. Kim, Y.-Y. Jung, A. W. Morales, and S.-J. Ko, "Spatio-temporal edge-based median filtering for deinterlacing," in *Consumer Electronics, 2000. ICCE. 2000 Digest of Technical Papers. International Conference on*. IEEE, 2000, pp. 52–53.

[12] M.Weston, "Interpolating lines of video signals," *US-patent 4,789,893*, December 1988.

[13] K. Lee and C. Lee, "High quality deinterlacing using content adaptive vertical temporal filtering," *Consumer Electronics, IEEE Transactions on*, vol. 56, no. 4, pp. 2469–2474, 2010.

[14] K. Lee and C. LEe, "High quality spatially registered vertical temporal filtering for deinterlacing," *Consumer Electronics, IEEE Transactions on*, vol. 59, no. 1, pp. 182–190, 2013.

[15] J. Wang, G. Jeon, and J. Jeong, "Deinterlacing algorithm with an advanced non-local means filter," *Optical Engineering*, vol. 51, no. 4, pp. 047 009–1, 2012.

[16] S.-M. Hong, S.-J. Park, J. Jang, and J. Jeong, "Deinterlacing algorithm using fixed directional interpolation filter and adaptive distance weighting scheme," *Optical Engineering*, vol. 50, no. 6, pp. 067 008–067 008, 2011.

[17] Q. Huang, D. Zhao, S. Ma, W. Gao, and H. Sun, "Deinterlacing using hierarchical motion analysis," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 20, no. 5, pp. 673–686, 2010.

[18] T. A. Ell and S. J. Sangwine, "Hypercomplex fourier transforms of color images," *Image Processing, IEEE Transactions on*, vol. 16, no. 1, pp. 22–35, 2007.

[19] C. M. Zwart, R. Venkatesan, and D. H. Frakes, "Decomposed multidimensional control grid interpolation for common consumer electronic image processing applications," *Journal of Electronic Imaging*, vol. 21, no. 4, pp. 043 012–043 012, 2012.

[20] R. Venkatesan, C. M. Zwart, and D. H. Frakes, "Video deinterlacing with control grid interpolation," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*. IEEE, 2012, pp. 861–864.

[21] P. Seeling, F. H. Fitzek, and M. Reisslein, *Video traces for network performance evaluation: a comprehensive overview and guide on video traces and their utilization in networking research*. Springer, 2007.

[22] D. M. Chandler and S. S. Hemami, "Vsnr: A wavelet-based visual signal-to-noise ratio for natural images," *Image Processing, IEEE Transactions on*, vol. 16, no. 9, pp. 2284–2298, 2007.

[23] H. R. Sheikh, A. C. Bovik, and G. De Veciana, "An information fidelity criterion for image quality assessment using natural scene statistics," *Image Processing, IEEE Transactions on*, vol. 14, no. 12, pp. 2117–2128, 2005.