Received August 6, 2019, accepted September 9, 2019, date of publication September 17, 2019,
date of current version September 27, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2942004

Efficient Edge-Based Image Interpolation Method Using Neighboring Slope Information

SAJID KHAN[1], DONG-HO LEE[2], MUHAMMAD ASIF KHAN[3], (Member, IEEE), ABDUL REHMAN GILAL[4], AND GHULAM MUJTABA[1]

[1]Center of Excellence for Robotics, Artificial Intelligence and Blockchain, Sukkur IBA University, Sukkur, Pakistan
[2]Division of Electrical Engineering, Hanyang University ERICA Campus, Ansan, South Korea
[3]Department of Electrical Engineering, Sukkur IBA University, Sukkur, Pakistan
[4]Department of Computer Science, Sukkur IBA University, Sukkur, Pakistan

Corresponding authors: Sajid Khan (sajidkhan@iba-suk.edu.pk) and Dong-Ho Lee (dhlee77@hanyang.ac.kr)

ABSTRACT This paper presents an image interpolation method that provides superior performance with low complexity. Applying a simple linear method achieves time-efficient interpolation, but often produces artifacts; however, applying other complex methods reduce unwanted artifacts at the cost of high computation time. The proposed interpolation scheme is based on the improvement of an algorithm called "Edge Slope Tracing (EST)" that predicts the slope based on the information of the adjacent slopes. Predicted slopes are used in slope-based line averaging to perform the interpolation. Slope-based line averaging is followed by post-processing of an interpolated image using two-way interpolation and thin edge correction to avoid the production of unwanted artifacts at the cost of low complexity. The proposed algorithm is very simple and simulation results indicate that it provides better or nearly equal results compared to conventional state-of-the-art algorithms and that it also reduces complexity to a great extent compared to conventional methods.

INDEX TERMS Edge, interpolation, slope tracing, super resolution, UDTV, upscaling.

I. INTRODUCTION

Images and videos exist in digital formats that are discrete representations of realistic scenarios. Every digital image has certain resolution, it will result in the blurring, jagging and distortion of the edges if someone tries to view that image beyond its resolution,. To deal with such problems, image interpolation is applied to upscale images to provide a pleasant viewing experience for low resolution images on high resolution devices.

Image interpolation is used in many applications and fields of image processing [1], [2]. One of the main applications of image interpolation is the upscaling of video. Since video consists of image frames, image interpolation is applied to every frame to upscale the video collectively. Video interpolation is needed for viewing low resolution video on high resolution TVs, as the TV industry has evolved from analog to digital, SDTV followed by HDTV and UDTV. Videos are not always taken with a good quality camera and viewed on an HD or UDTV; rather, many times low-resolution classical videos are seen over HD/UDTV. Therefore, in such situations, interpolation is used to upscale a video to be viewed on higher resolution TVs.

Image interpolation is used in various applications such as medical image processing [3], magnetic resonance imaging (MRI), digital subtraction angiography (DSA), computer-aided diagnosis (CAD), computer-assisted surgery (CAS), and picture archiving and communication systems (PACS). These applications use image interpolation either to zoom or rotate the medical images [4]. Image interpolation is also used in gaming engines to provide users pleasant experience while playing classical games.

The objective of a good interpolation algorithm is to enhance the resolution of an image or video frame while preserving image details such as edges. A good interpolation algorithm needs to be simple and must have low complexity, so that it can easily be implemented. Many researchers [5]–[18] have proposed algorithms for upscaling digital images. Nearest neighbors, bilinear and bicubic [4] are popular and time-efficient interpolations. However, these all (i.e., nearest neighbors, bilinear and bicubic) result in the blurring and smoothing of transition regions and edges which causes unpleasant experience, since human eyes are sensitive to these regions.

The associate editor coordinating the review of this manuscript and approving it for publication was Yong Yang.

Paik *et al.* [5] used cross-correlations of horizontal and vertical directions to find the maximum cross-correlation point for interpolation. Allebach and Wong [6] proposed two-stage interpolation by applying bilinear interpolation-based rendering followed by data correction along edges. Chen *et al.* [7] classified pixels into complex and smooth regions after a comparison of the local pixel gradient with a threshold and applied trilateral and bilateral filters to complex and smooth regions, respectively. Dong and Zhang [8] used nonlocal autoregressive modeling with a combination of sparse representation modeling. Shi and Shan [9] proposed an approach based on the processing of a 7×7 window. They defined six modes; each mode corresponds to a specific angle. They assigned pixel locations to each mode and calculated the summation of absolute differences of those locations for every mode. Those algorithms still fail in precise restoration of edges or blur images that contain textures or edges.

Yang *et al.* [11] proposed dictionary learning based interpolation method Super Resolution via Sparse Representation (SRSR) that provides pleasant results in upscalling images. However, its complexity makes it inconvenient for real time processing.

Li and Orchard [12] proposed a state-of-the-art interpolation method named New Edge-Directed Interpolation (NEDI). NEDI is a covariance-based adaptive interpolation technique that approximates the covariance of a high-resolution image on the basis of its correspondence with low-resolution covariance. NEDI provides high performance at the cost of very high complexity. Asuni and Giachetti [13] proposed improved NEDI (iNEDI), which produced better performance than NEDI. Both NEDI and iNEDI are state-of-the-art algorithms, but their high complexity is still an issue.

Giachetti and Asuni [14], [15] proposed Iterative Curvature-based Interpolation (ICBI), which provides considerably improved results compared to conventional methods. ICBI first calculates second-order derivatives along the two diagonal directions and the diagonal that has the minimum second order derivative is used for interpolation. Interpolated values are modified again in the second step with the help of the minimization of an energy function. ICBI exhibits much better performance and lower complexity compared to NEDI and iNEDI, but ICBI is not simple enough to be easily implemented since a high complexity problem isn't solved to a considerable extent.

Zhao *et al.* [16], Timofte *et al.* [17], and Zhu *et al.* [18] have proposed dictionary learning based image interpolation methods named Self-learning and Adaptive Sparse Representation (SL-ASR), Anchored Neighborhood Regression (ANR) and Single Image Super Resolution (SISR), respectively. SISR uses multiple dictionaries that make it far more complex method compared to ANR and SL-ASR. Problem with dictionary learning based interpolation algorithms is that they sometimes change property of a region completely by either introducing some artifacts or changing intensities of a region using the patches available in the dictionary.

In this paper, a new interpolation algorithm is proposed which is based on the proposed technique Extended Edge Slope Tracing (EEST). EEST predicts the slope of the current pixel on the basis of the information obtained from the slope of the adjacent pixel. EEST is based on improvement of EST that was first introduced for deinterlacing [19], [20]. It has some serious problems with interpolation due to limited slope candidates and using of 1×3 window based artifacts correction techniques that depends on results of line averaging which causes in blurring along the edges when used for interpolation. The proposed method provides better slope calculation and reset criteria. It also uses efficient artifact reduction and correction techniques consisting of two-way interpolation, and thin line correction. The complexity of the proposed algorithm is closer to linear filters and it is far less complex if compared to NEDI, ICBI and SL-ASR as there is no need to calculate any coefficients or any iterative processes. Simulation results show that proposed algorithm provides better results compared to NEDI and SL-ASR whereas compared to ICBI it provides similar results in some regions and better results in others.

The rest of this paper is comprised of five section. Some of the conventional methods are described in the following section II. Section III describes the proposed EEST-based interpolation algorithm. The performance analysis of the algorithm is discussed explicitly in the Section IV. At the end, conclusion of the study is presented in the Section V.

II. NEDI [12], ICBI [14] AND SL-ASR [17]

NEDI [12] is based on an assumption that a low-resolution image is extracted from a high-resolution image with $I(2i, 2j) = X_{i,j}$. X is the input low-resolution image, whereas I is the high-resolution output image. Since pixels with both even indices in I are pixels of the low-resolution image, pixels at odd locations can be calculated using (1).

$$I_{(2i+1,2j+1)} = \sum_{k=0}^{1}\sum_{l=0}^{1} \alpha_{(2k+1)} I_{(2(i+k),2(j+l))} \tag{1}$$

The coefficients are given in (2):

$$\overrightarrow{\alpha} = R^{-1}\overrightarrow{r} \tag{2}$$

where $R = [R_{kl}]$ and $\overrightarrow{r} = [r_k]$ are the local covariance at the high resolution shown in Figure 1 and can be calculated in (3).

$$R = \frac{1}{M^2}C^T C, \quad \overrightarrow{r} = \frac{1}{M^2}C^T\overrightarrow{y} \tag{3}$$

where $\overrightarrow{r} = [y_1, y_2, \ldots\ldots, y_k, \ldots. y_{M^2}]^T$ is the data vector containing $M \times M$ pixels inside the local window and C is a $4 \times M$ data matrix whose k^{th} column vector is the four nearest neighbors of y_k along the diagonal direction. Putting the values of R and $\overrightarrow{y}$ in (2) results in the final equation of interpolation coefficients:

$$\overrightarrow{\alpha} = (C^T C)^{-1} C^T \overrightarrow{y} \tag{4}$$

In a similar way, pixels that don't have both odd indices are interpolated in the second step. The vectors used in NEDI can be visualized from Figure 1.

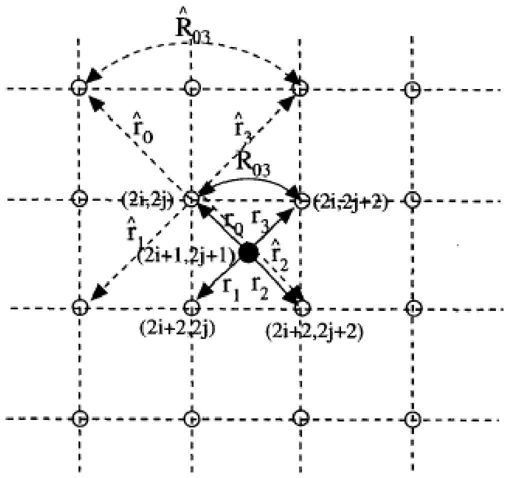


FIGURE 1. Correspondence between low and high-resolution covariance.

ICBI [14] first interpolates pixels along with any of the two diagonals on the basis of the minimum response along the diagonal, and then it modifies interpolated values further using an iterative procedure that minimizes an energy function. The objective of energy function is to smooth the blurred edges obtained after the first step of interpolation. The energy function that is subjected to minimization is given in (5):

$$U(2i+1,2j+1) = aU_d(2i+1,2j+1)+bU_e(2i+1,2j+1)+cU_i(2i+1,2j+1) \quad (5)$$

where U_d, U_e, and U_i are calculated using (6), (7) and (8):

$$U_d(i,j) = w_1D_1 + w_2D_2 + w_3D_3 + w_4D_4 \quad (6)$$
$$U_e(i,j) = |I_{11}(\text{i,j})| + |I_{22}(\text{i,j})| \quad (7)$$
$$U_i(i,j) = f(I)|_{(i,j)}\, I(\text{i},j) \quad (8)$$

where

$$D_1 = |I_{11}(\text{i},j) - I_{11}(\text{i}+1,\text{j}+1)|+|I_{22}(i,j)-I_{22}(\text{i}+1,j+1)| \quad (9)$$

$$D_2 = |I_{11}(\text{i},j) - I_{11}(\text{i}+1,j-1)|+|I_{22}(\text{i},j)-I_{22}(\text{i}+1,j-1)| \quad (10)$$

$$D_3 = |I_{11}(\text{j},j) - I_{11}(\text{i}-1,j+1)|+|I_{22}(\text{i},j)-I_{22}(\text{i}-1,j+1)| \quad (11)$$

$$D_4 = |I_{11}(\text{i},j) - I_{11}(\text{i}-1,j-1)|+|I_{22}(\text{i},j)-I_{22}(i-1,j-1)| \quad (12)$$

$$f(I) = \frac{I_1(i,j)^2 I_{22}(i,j)}{I_1(i,j)^2+I_2(i,j)^2} - \frac{2I_1(i,j)I_2(i,j)I_{12}(i,j)-I_{11}(i,j)^2I_2(i,j)}{I_1(i,j)^2+I_2(i,j)^2} \quad (13)$$

where I_1, I_2, I_{11}, I_{22} and I_{12} are local approximations of first and second order differentials.

SL-ASR [17] is a patch based dictionary learning approach that divides the low-resolution input image into n overlapped patches $x_i(x_i \in \{R^m|i = 1,2,\ldots,n\})$ in image X with locations i. It then takes a dictionary $D_h = [d_1, d_2, \ldots., d_k] \in R^{m\times k}$ and obtain an approximation of input patch from the dictionary D_h based on minimization of function in (14)

$$\min_{\alpha_i} ||x_i\text{-D}_h\alpha_i||_2^2 + \lambda_i||\alpha_i|| \quad (14)$$

where $\alpha_i \in R^{kl}$ is a sparse vector and $\lambda > 0$ is a parameter that balances the trade-off between fidelity term and sparsity priors.

III. PROPOSED METHOD

The proposed algorithm upscale the images by interpolating along the row direction and then along the column direction. Both row and column interpolation follow the same procedure. An original low-resolution image of size $M \times N$ is used as the input image for interpolation along the row direction to obtain row interpolated image I^{Row} of size $2M \times N$. The transpose of I^{Row} is then used as the input image for column interpolation to obtain an interpolated image of size $2N \times 2M$. The interpolated image of size $2N \times 2M$ is the final interpolated image, but its transpose is needed to obtain a final interpolated image I of size $2M \times 2N$.

The proposed algorithm is an image interpolation method with high performance and low complexity compared to other state-of-the-art algorithms such as NEDI, ICBI and SL-ASR. The proposed algorithm is based on a simple, efficient technique, EEST, which predicts the present slope change by using the previous slope information. The slope value obtained by EEST is used for slope-based interpolation. To remove unwanted artifacts, correction techniques: two-way interpolation and thin edge correction are applied at the end. Before going into the detail on the proposed algorithm, EEST is explained in detail.

A. EXTENDED EDGE SLOPE TRACING

The flow chart of the entire EEST process is shown in Figure 2. Slope k_{cur} of the current pixel is calculated on the basis of the extreme left, left, right, extreme right and middle slopes calculated by using the slope k_{prev} (the slope of the adjacent pixel). The middle (S_{mid}), extreme left (S_{Eleft}), left (S_{left}), right (S_{right}) and extreme right (S_{Eright}) slope

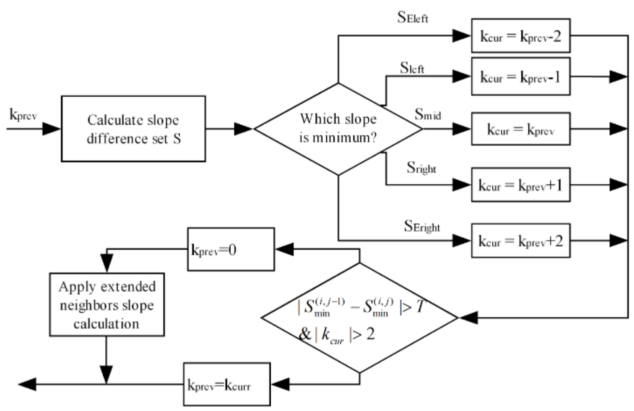


FIGURE 2. Flowchart of the Extended Edge Slope Tracing (EEST) process.

differences that are used in calculating k_{cur} are given in (15)

$$S = \begin{cases} S_{mid} = \left|I_{(i-1,j+k_{prev})} - I_{(i+1,j-k_{prev})}\right| \\ S_{left} = \left|I_{(i-1,j+k_{prev}-1)} - I_{(i+1,j-k_{prev}+1)}\right| \\ S_{right} = \left|I_{(i-1,j+k_{prev}+1)} - I_{(i+1,j-k_{prev}-1)}\right| \\ S_{Eright} = \left|I_{(i-1,j+k_{prev}+2)} - I_{(i+1,j-k_{prev}-2)}\right| \\ S_{Eleft} = \left|I_{(i-1,j+k_{prev}\text{-}2)} - I_{(i+1,j-k_{prev}+2)}\right| \end{cases} \tag{15}$$

The three examples for calculating the middle, right, left, extreme right and extreme left slopes using different k_{prev} values are shown in Figure 3. For all the calculations, it is assumed that the current pixel is at the center of the patches shown for each case. Current slope k_{cur} is calculated by incrementing, decrementing or using the same value of the previous slope k_{prev} depending on the conditions given in (16).

$$\begin{cases} k_{cur} = k_{prev} - 2, & \text{if } \min(S) = S_{Eleft} \\ k_{cur} = k_{prev} - 1, & \text{if } \min(S) = S_{left} \\ k_{cur} = k_{prev} + 1, & \text{if } \min(S) = S_{right} \\ k_{cur} = k_{prev} + 2, & \text{if } \min(S) = S_{Eright} \\ k_{cur} = k_{prev}, & \text{else} \end{cases} \tag{16}$$

To avoid the production of unwanted artifacts in case a wrong value exists for the reference to previous slope k_{prev}, resetting criteria is used by resetting the value of the reference previous slope to 0 if the condition of $\left(\left|S_{\min}^{(i,j)} - S_{\min}^{(i,j-1)}\right| > T \,\&\, |k_{cur}| > 2\right)$ is satisfied. A large value of $\left|S_{\min}^{(i,j)} - S_{\min}^{(i,j-1)}\right|$ shows that the edge in that region is not smooth anymore or discontinued due to the beginning of a new edge and it is not possible to predict the current slope on the basis of the information of the previous slope.

Large value of T results in failure of efficient reset of reference previous slope, whereas small value of T results in frequent reset. The value of $T = 15$ is used on the basis of extensive simulations. After setting previous slope to 0,

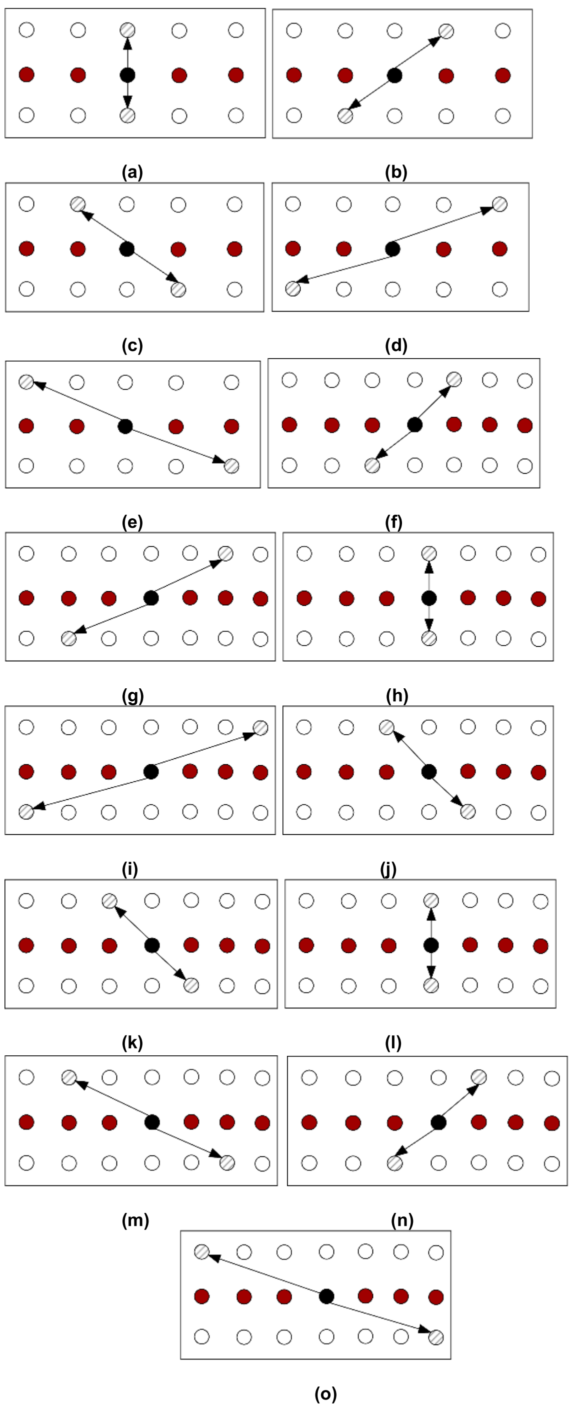


FIGURE 3. Examples of middle, right, left, extreme right and extreme left slope calculation when previous slope is: (a)-(e) 0, (f)-(j) 1, (k)-(o) −1.

the current slope k_{cur} is calculated using (17)

$$\begin{cases} k_{cur} = -3, & \text{if } \min(S) = S_{Eleft-1} \\ k_{cur} = -2, & \text{if } \min(S) = S_{Eleft} \\ k_{cur} = -1, & \text{if } \min(S) = S_{left} \\ k_{cur} = 1, & \text{if } \min(S) = S_{right} \\ k_{cur} = 2, & \text{if } \min(S) = S_{Eright} \\ k_{cur} = 3, & \text{if } \min(S) = S_{Eright+1} \\ k_{cur} = 0, & \text{else} \end{cases} \tag{17}$$

where S_{Eleft}, S_{left}, S_{mid}, S_{right} and S_{Eright} can be calculated using (15) using $k_{prev} = 0$ and $S_{Eleft-1}$ and $S_{Eright+1}$ can be calculated using (18) and (19)

$$S_{Eleft-1} = \left|I_{(i-1,j-3)} - I_{(i+1,j+3)}\right| \quad (18)$$
$$S_{Eright+1} = \left|I_{(i-1,j+3)} - I_{(i+1,j-3)}\right| \quad (19)$$

B. SLOPE-BASED INTERPOLATION

On the basis of slope k_{cur} calculated using EEST, the pixel at location (i, j) can be interpolated using (20).

$$I(i,j) = \frac{I_{(i-1,j+k_{cur})} + I_{(i+1,j-k_{cur})}}{2} \quad (20)$$

Figure 4 shows different examples of the calculation of the interpolated pixel at location (i, j) using different values of k_{cur}. Circles with thick boundaries show pixels that are used in slope-based interpolation whereas centering filled circles show pixels that are interpolated by averaging small gray pixels.

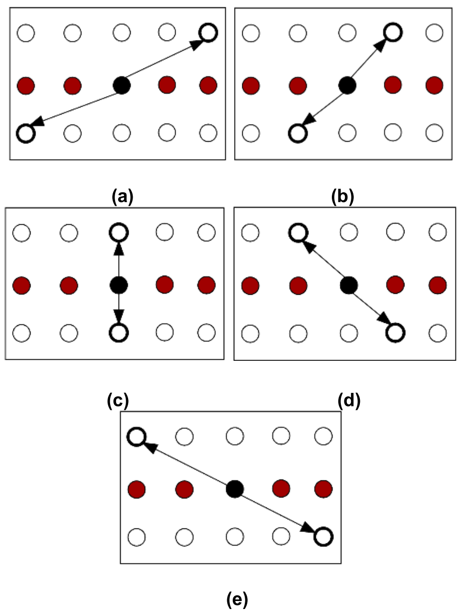


FIGURE 4. Examples of slope-based interpolation using different slopes: (a) 2, (b) 1, (c) 0, (d) −1, and (e) −2.

C. THE WHOLE ALGORITHM WITH ARTIFACT REDUCTION

The process of the entire proposed algorithm is shown in Figure 5. It is observed that for a slope of 90° or ±75°, line averaging works well, whereas for slopes other 90° or ±75°, slope-based interpolation works well and produces good results on the basis of slope prediction. The decision to apply line averaging or slope-based interpolation can be made using (21), (22) and (23).

$$\begin{aligned} d_1 &= \left|I_{(i-1,j-1)} - I_{(i+1,j-1)}\right| + \left|I_{(i-2,j)} - I_{(i+1,j)}\right| \\ &\quad + \left|I_{(i-1,j+1)} - I_{(i+1,j+1)}\right| \quad (21) \\ d_2 &= \left|I_{(i-1,j-1)} - I_{(i+1,j)}\right| + \left|I_{(i-1,j)} - I_{(i+1,j+1)}\right| \quad (22) \\ d_3 &= \left|I_{(i-1,j)} - I_{(i+1,j-1)}\right| + \left|I_{(i-1,j+1)} - I_{(i+1,j)}\right| \quad (23) \end{aligned}$$

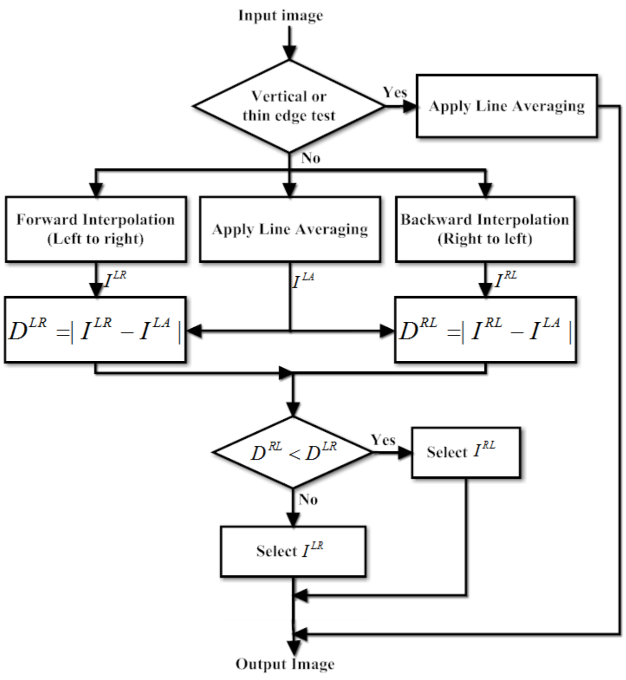

Input image
Vertical or thin edge test
Yes
Apply Line Averaging
No
Forward Interpolation (Left to right)
Apply Line Averaging
Backward Interpolation (Right to left)
I^{LR}
I^{LA}
I^{RL}
$D^{LR} = |I^{LR} - I^{LA}|$
$D^{RL} = |I^{RL} - I^{LA}|$
$D^{RL} < D^{LR}$
Yes
Select I^{RL}
No
Select I^{LR}
Output Image

FIGURE 5. Flow chart of the proposed interpolation method.

Figure 6 shows the calculation of d_1, d_2 and d_3 in (a), (b) and (c) respectively. Line averaging is applied to the pixel at location (i, j) if the condition of $\min(d_1, d_2, d_3) \leq Th$ is satisfied. Small value of Th decreases complexity of the algorithm since line averaging is applied to more pixels but also results in jagging. An optimal value of $Th = 20$ is used on the basis of extensive simulations. In other locations where $\min(d_1, d_2, d_3) > Th$, slope-based interpolation is applied.

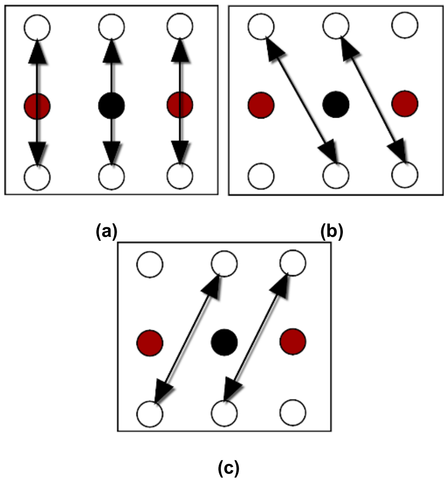


FIGURE 6. Differences for determining ±75 or 90 degree slope; (a) d1, (b) d2, and (c) d3.

Slope-based interpolation sometimes destroys lines or edges that are very thin. If a very thin edge separates two uniform regions that have similar intensity values, in such cases few pixels from the thin edges may be subjected to distortion, since there is a chance that the difference among the pixels in those two uniform regions appears to be smaller compared to the difference along a thin edge. In such cases, slope-based interpolation selects pixels from those two uniform regions for interpolation. To avoid such failure, if at least three differences in S appear to be smaller than the threshold, then line averaging is applied. Hence, if a pixel is at 90° or at ±75° or the edge is thin, line averaging is applied. Slope-based interpolation is applied to the rest of the regions.

Pixels that don't have a 90° or ±75° slope or are not detected as thin edge pixels are interpolated along both the left to right (forward interpolation) and right to left (backward interpolation) directions to get I^{LR} and I^{RL}, respectively. I^{LR} and I^{RL} are required for the removal of any unwanted artifacts produced. The decision to use the interpolated value from right to left or left to right interpolation is based on differences, as in (24) and (25).

$$D^{LR} = \left| I^{LR} - I^{LA} \right| \tag{24}$$

$$D^{RL} = \left| I^{RL} - I^{LA} \right| \tag{25}$$

On the basis of D^{LR} and D^{RL}, final interpolated image I is obtained by using interpolated values in I^{LR} that correspond to small differences in D^{LR} compared to D^{RL} and vice versa.

IV. PERFORMANCE EVALUATION

The performance of the proposed algorithm is evaluated by applying bicubic, NEDI, ICBI, SL-ASR and the proposed algorithm to 25 grayscale images provided in [21] and 5 selected color images provided in [22]. Patch size for SL-ASR used is 256. This study performed both subjective and objective tests in order to compare quantitatively the quality of the images created with different methods and the related computational costs. MATLAB 2018a was used for collecting all results and the tic toc function of MATLAB was used to calculate the execution time.

A. OBJECTIVE ANALYSIS

For statistical calculations, test images given in [21], [22] are reduced by factors of 2 and 4 for objective evaluation. Images interpolated by a factor of 2 or 4 are compared with the original images. The PSNR, as presented in (26), is used for objective comparisons with other methods.

$$PNSR_I = 10 \log_{10} \left(\frac{Max_I^2}{MSE_I} \right) \tag{26}$$

where I is an upscaled image and MSEI is the mean square error of I. Table 1 shows the average PSNR for 25 grayscale test images along with 10 color test images when upscaled by a factor of 2 and 4 using bicubic, NEDI, ICBI, SL-ASR, and the proposed method. From the table, it can be observed that ICBI provides the highest PSNR in all cases. When upscaled by 2, ICBI is much higher than the existing methods: around 3 dB higher than bicubic and around 1 dB higher than NEDI. The PSNR of the proposed method is little smaller than ICBI. When upscaled by 4, ICBI is almost 2 dB higher than bicubic and 1 dB higher than NEDI. The proposed method provides similar PSNR to ICBI. PSNR of SL-ASR is lowest

TABLE 1. Average PSNR (dB) comparison when upscaled by 2× and 4× factors.

Scaling factor	Bicubic	NEDI	ICBI	SL-ASR	Proposed
2x	26.15	28.60	29.14	23.65	28.95
4x	23.53	24.95	25.57	19.12	25.46
Color image 2x	26.86	28.51	29.50	24.11	29.16

TABLE 2. Average elapsed CPU time (sec.) when upscaled by 2× and 4× factors.

Scaling factor	Bicubic	NEDI	ICBI	SL-ASR	Proposed
2x	0.008	4.792	1.443	4.044	0.017
4x	0.039	23.104	7.334	20.867	0.097
Color image 2x	0.067	39.915	11.554	29.373	0.120

TABLE 3. Operation units for implementation.

Methods	No. of Line Memory	No. of Adders	No. of Multipliers	No. of Comparators
Bicubic	4	16	16	0
Proposed	2	29	0	12

FIGURE 7. Selected test images for the subjective evaluations.

because it provides interpolation by selecting a patch from its input dictionary. In most of the cases, selected patch does not completely match local features of original image.

Table 2 shows a comparison of the execution time for all five algorithms when they were applied to the test images given in [19], [20]. The execution time for bicubic is far less compared to NEDI and ICBI. Bicubic is 2-2.5 times faster compared to the proposed algorithm. On average, the proposed algorithm is 240-280 times faster compared to NEDI, 215-238 times faster than SL-ASR, and 76-84 times faster than ICBI. For color images, the proposed algorithm

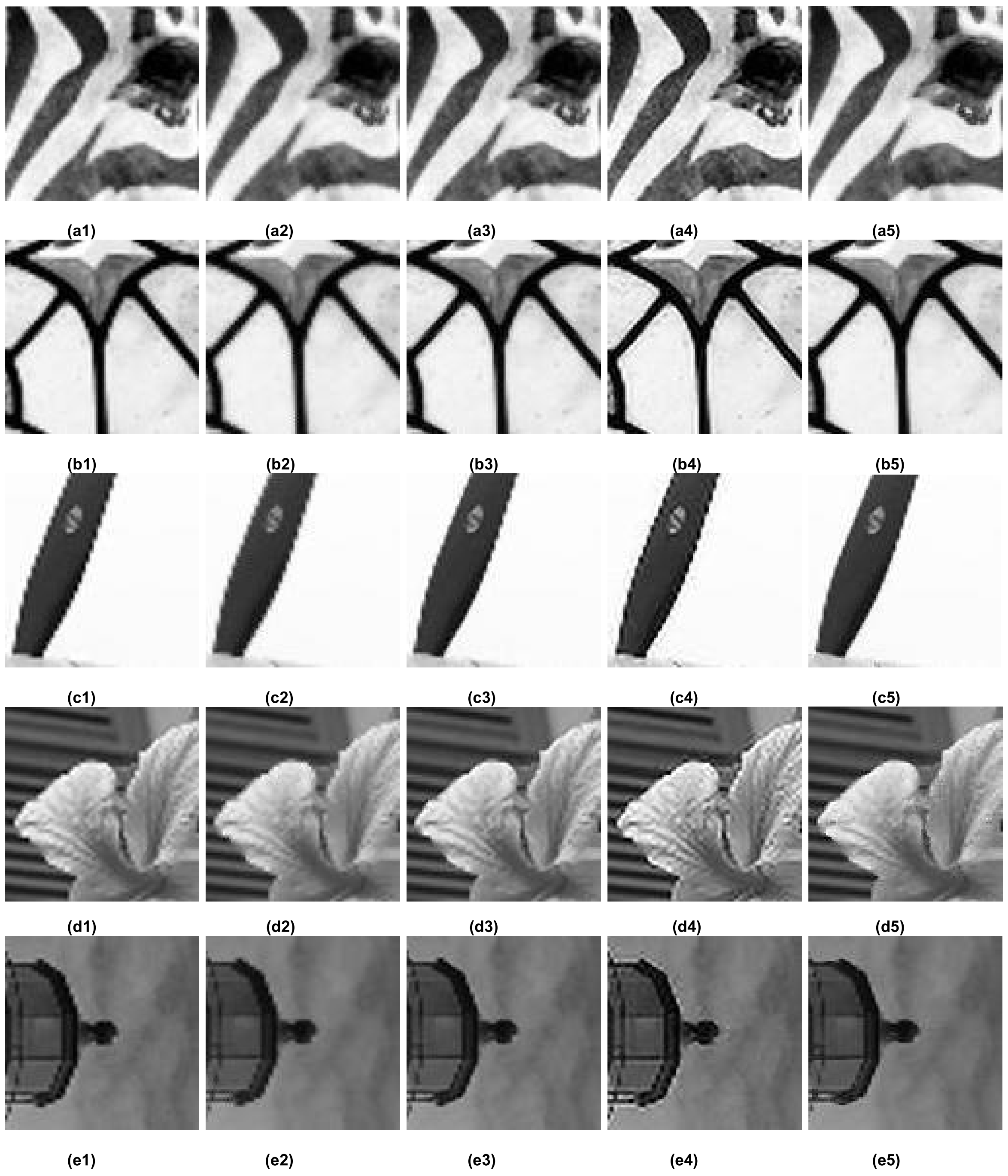


FIGURE 8. Cropped regions after upscaling (×2): (a1)-(e1) Bicubic, (a2-e2) NEDI, (a3-e3) ICBI, (a4-e4) SL-ASR, and (a5-e5) Proposed.

is 1.8 times slower compared to bicubic, 332 times faster compared to NEDI, 96 times faster than ICBI, and 245 times faster than SL-ASR.

Since predefined function for Bicubic interpolation is implemented by using mex compiler in MATLAB, therefore, comparing the proposed method with Bicubic interpolation on the basis of computation of execution time is not perfect a criterion. Table 3 shows the operation units which should be considered during the implementation. Since it is useless to compute the units for NEDI, ICBI and SL-ASR, the table does not include the units for NEDI, ICBI and SL-ASR. The proposed algorithm is only compared with the bicubic methods, where the proposed method requires just two line memories; bicubic requires 4 line memories. Bicubic requires 16 adders, and the proposed method using EEST requires 29. The proposed method does not require a multiplier, and bicubic requires 16 multipliers. The proposed method requires 12 comparators whereas only 1 comparator is required for the case of line averaging. From the table, with respect to hardware implementation, the proposed algorithm is much simpler than the bicubic method.

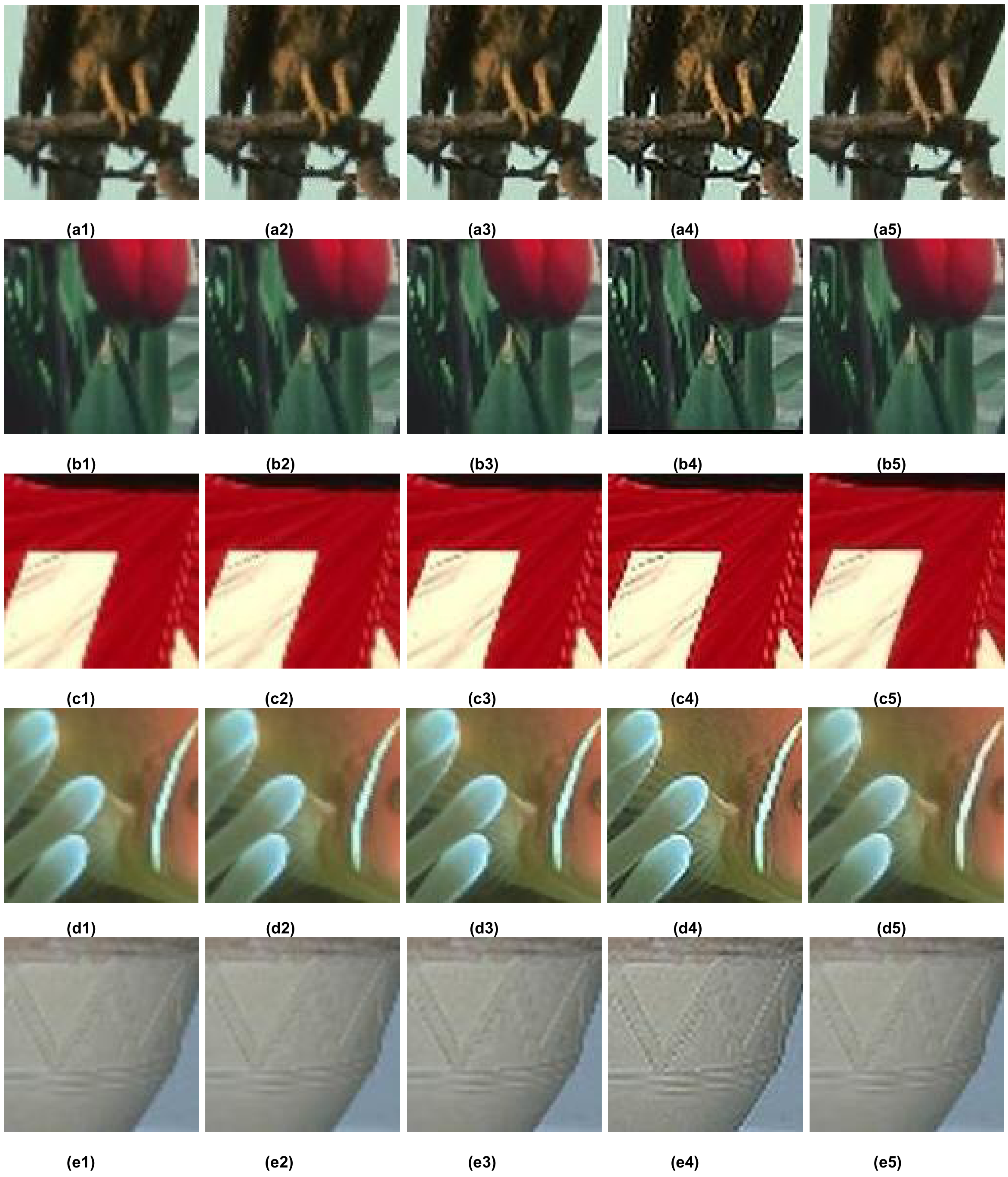


FIGURE 9. **Cropped regions after upscaling (×2): (a1)-(e1) Bicubic, (a2-e2) NEDI, (a3-e3) ICBI, (a4-e4) SL-ASR, and (a5-e5) Proposed.**

B. SUBJECTIVE ANALYSIS

Figure 7 shows a set of images selected for subjective evaluation because they represent a variety of patterns. The proposed algorithm can be applied to both color and grayscale images. The proposed algorithm can be applied to color images by either treating each red, green and blue channel of a RGB image individually, however, it is recommended to calculate slopes for one channel and using the same slopes for the other two channels to reduce the execution time. The performance of the interpolation algorithm can be judged by performing an in-depth analysis of the image.

Figure 8 shows some cropped regions of all grayscale test images that are subjected to be upscaled by a factor of two. It can be seen that bicubic, NEDI, and SL-ASR failed to restore edges distorted due to downscaling the original images. Hence, producing jagging along the edges, whereas ICBI and the proposed algorithm restored edges efficiently compared to other algorithms. For the case of the zebra (a) and glass window (b) images, jagging and blurring along the edges are clear for the application of bicubic, NEDI, and SL-ASR. They failed to restore edges with smooth slopes. For the case of the airplane image (c) and lighthouse image (e), the proposed algorithm produces better results for some cropped regions compared to ICBI.

Figure 9 shows some cropped regions of all color test images upscaled by a factor of 2. Similar to the results of grayscale images, ICBI and the proposed algorithms preserved edges efficiently, whereas bicubic and NEDI produced jagging along edges. SL-ASR also produced jagging and artifacts along edges, and increased the intensity of smooth regions.

For the cases of eagle image (a), horse race image (c), and vase image (e), jagging along edges especially along smooth edges is clear for bicubic, NEDI, and SL-ASR. SL-ASR. However, they have produced some pleasant results but also produced some artifacts along edges. ICBI and the proposed algorithms, however, performed very well compared to bicubic, NEDI, and SL-ASR since they restore edges and other image details clearly, as there was no jagging or blurring along the edges and other image details are preserved more efficiently. For all cases, the proposed algorithm produces similar performance compared to ICBI and, in some cases, the proposed algorithm restored details and edges more efficiently compared to ICBI. Proposed algorithm provided similar results compared to ICBI for eagle (a), and building (b) images whereas slightly better compared to ICBI for horse race (b), fish (d), and vase (e) images. For all colored images, slopes were calculated for green channel only and same slopes were applied to red and blue channels to further increase time efficiency.

V. CONCLUSION

In this paper, an efficient edge-based interpolation method is proposed. A new technique, EEST, is introduced and used for interpolation that predicts the present slope on the basis of the information of previous slopes. The proposed interpolation offers a high performance at very low complexity. Moreover, to improve the accuracy of EEST-based interpolation, slope-resetting criteria are applied along with the application of two-way EEST-based interpolation and compensation for thin lines or edges. Simulation results showed that restorations of edges are clear in the proposed algorithm if compared to most of the state-of-the-art conventional algorithms. Moreover, the simple interpolation of an image without finding coefficients or without requiring large correlation masks unlike other conventional algorithms proves the superiority of the proposed algorithm on the basis of low complexity.

REFERENCES

[1] Y. Shishikui and Y. Sawahata, "Effects of viewing ultra-high-resolution images with practical viewing distances on familiar impressions," *IEEE Trans. Broadcast.*, vol. 64, no. 2, pp. 498–507, Jun. 2018.

[2] A. Amanatiadis and I. Andreadis, "A survey on evaluation methods for image interpolation," *Meas. Sci. Technol.*, vol. 20, no. 10, Sep. 2009, pp. 316–324.

[3] H. Liu, Q. Guo, G. Wang, B. B. Gupta, and C. Zhang, "Medical image resolution enhancement for healthcare using nonlocal self-similarity and low-rank prior," *Multimedia Tools Appl.*, vol. 78, no. 7, pp. 9033–9050, Apr. 2019.

[4] R. C. Gonzalez, *Digital Image Processing*, 3rd ed. Upper Saddle River, NJ, USA : Prentice-Hall, 2009.

[5] J. H. Paik, K. P. Hong, J. K. Paik, and J. H. Hwang, "Image sequence interpolation for improving the resolution of the magnified image," in *Proc. IEEE Asia Pacific Conf. Circuits Syst.*, Seoul, South Korea, Nov. 1996, pp. 544–547.

[6] J. Allebach and P. W. Wong, "Edge-directed interpolation," in *Proc. Int. Conf. Image Process.*, Lausanne, Switzerland, Sep. 1996, pp. 707–710.

[7] X. Chen, G. Jeon, and J. Jeong, "Filter switching interpolation method for deinterlacing," *Opt. Eng.*, vol. 51, no. 10, Oct. 2012, Art. no. 107402.

[8] W. Dong, L. Zhang, R. Lukac, and G. Shi, "Sparse representation based image interpolation with nonlocal autoregressive modeling," *IEEE Trans. Image Process.*, vol. 22, no. 4, pp. 1382–1394, Apr. 2013.

[9] J. Shi and Z. Shan, "Image interpolation using a variation-based approach," in *Proc. Int. Conf. Inf., Commun. Signal Process.*, Singapore, Dec. 2011, pp. 1–4.

[10] Y.-L. Chang, S.-F. Lin, and L.-G. Chen, "Extended intelligent edge-based line average with its implementation and test method," in *Proc. IEEE Int. Symp. Circuits Syst.*, Vancouver, BC, Canada, May 2004, pp. 341–344.

[11] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.

[12] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE Trans. Image Process.*, vol. 10, no. 10, pp. 1521–1527, Oct. 2001.

[13] N. Asuni and A. Giachetti, "Accuracy improvements and artifacts removal in edge based image interpolation," in *Proc. Int. Conf. Comput. Vis. Theory Appl.*, Funchal, Portugal, Jan. 2008, pp. 352–355.

[14] A. Giachetti and N. Asuni, "Real-time artifact -free image upscaling," *IEEE Trans. Image Process.*, vol. 20, no. 10, pp. 2760–2768, Oct. 2011.

[15] A. Giachetti and N. Asuni, "Corrections to 'Real-time artifact free image upscaling,'" *IEEE Trans. Image Process.*, vol. 21, no. 4, p. 2361, Apr. 2012.

[16] J. Zhao, T. Sun, and F. Cao, "Image super-resolution via adaptive sparse representation and self-learning," *IET Comput. Vis.*, vol. 12, no. 5, pp. 753–761, 2018.

[17] R. Timofte, V. De Smet, and L. Van Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1920–1927.

[18] X. Zhu, X. Wang, J. Wang, P. Jin, L. Liu, and D. Mei, "Image super-resolution based on sparse representation via direction and edge dictionaries," *Math. Problems Eng.*, vol. 2017, Jun. 2017, Art. no. 3259357.

[19] S. Khan and D. H. Lee, "Efficient deinterlacing method using simple edge slope tracing," *Opt. Eng.*, vol. 54, no. 10, Oct. 2015, Art. no. 103108.

[20] D.-H. Lee, "A simple, high performance edge-adaptive deinterlacing algorithm with very low complexity," in *Proc. IEEE Int. Conf. Consum. Electron.*, Las Vegas, NV, USA, Jan. 2012, pp. 636–637.

[21] *ICBI-Iterative Curvature Based Interpolation*. Accessed: Jan. 8, 2019. [Online]. Available: http://www.andreagiachetti.it/icbi/TEST_IMAGES.tar.gz

[22] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 2, Jul. 2001, pp. 416–423.

SAJID KHAN was born in Sukkur, Pakistan, in 1988. He received the B.S. degree in telecom engineering from FAST-NUCES University, Pakistan, in 2011, and the M.S. leading to Ph.D. degrees in electronics and communication engineering from Hanyang University, Ansan, South Korea, in 2017.

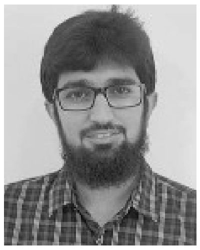
From 2011 to 2012, he was a Software Engineer with Gameview Studios, Pakistan. Since 2017, he has been as an Assistant Professor with the Computer Science Department, Sukkur IBA University, Sukkur. He is the author of two articles whereas coauthor of two other articles. His research interests include image denoising, edge detection, interpolation, deinterlacing, fingerprint detection, and biomedical image processing. Dr. Khan is a member of Pakistan Engineering Council. He was awarded scholarships for B.S. and M.S. leading Ph.D. by ministry of ICT Pakistan and HEC, respectively.

DONG-HO LEE received the B.S. degree in electronic engineering from Hanyang University, Seoul, South Korea, in 1986, and the M.S. and Ph.D. degrees in electrical and computer engineering from The University of Texas at Austin, in 1988 and 1991, respectively.

From 1991 to 1994, he was a Senior Engineer with LG Electronics involving the development and implementation of DTV systems. Since 1994, he has been a Professor with the Department of Electronics and Communication Engineering, Hanyang University ERICA Campus, Ansan, South Korea. His research interests include digital image processing and pattern recognition.

MUHAMMAD ASIF KHAN was born in Sukkur, Pakistan, in 1984. He received the Bachelor of Engineering degree in computer systems from QUEST University, Pakistan, in 2007, and the M.S. degree from Sukkur IBA University, in 2012. He is currently pursuing the Ph.D. degree in U.K.

From 2007 to 2013, he was a Fiber Optics Operation Engineer with Mobilink, Pakistan. He has been employed as an Assistant Professor with the Department of Electrical Engineering, Sukkur IBA University, since 2013. He is the author of one articles whereas coauthor of two other articles. His research interests include computer vision for Robots navigation, edge detection, and interpolation.



ABDUL REHMAN GILAL received the Ph.D. degree in information technology from Universiti Teknologi Petronas, Malaysia. He is currently a Faculty Member with the Computer Science Department, Sukkur IBA University, Pakistan. He has been mainly researching in the field of software project management for finding the effective methods of composing software development teams. Based on his research publication track record, he has contributed in the areas of human factor in software development, complex networks, image processing, databases and data mining, and programming and cloud computing.

GHULAM MUJTABA received the master's degree in computer science from FAST National University, Karachi, Pakistan, and the Ph.D. degree from the Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia. He has received the gold medal for the master's degree. Before, he joined Sukkur IBA University, he was with a well-known software house in Karachi for four years. He has been an Assistant Professor with Sukkur IBA University, Sukkur, Pakistan, since 2006. He has vast experience in teaching and research. He has also published several articles in academic journals indexed in well-reputed databases, such as ISI and Scopus. His research interests include machine learning, online social networking, text mining, deep learning, and information retrieval.