Regular Article

Dynamic Texture Map Based Artifact Reduction for Compressed Videos

Thai Van Nguyen[1], Tuan Do-Hong[1], Dung Trung Vo[2]

[1] Faculty of Electrical and Electronics Engineering, Ho Chi Minh City University of Technology, Vietnam National University Ho Chi Minh City, Vietnam
[2] Samsung Research America, Irvine, CA, USA 92626

Correspondence: Thai Van Nguyen, thai1279@gmail.com
Communication: received 31 March 2019, revised 7 August 2019, accepted 15 August 2019
Online publication: 23 November 2019, Digital Object Identifier: 10.21553/rev-jec.232
The associate editor coordinating the review of this article and recommending it for publication was Prof. Nguyen Le Hung.

Abstract– **This paper proposes a method of artifact reduction in compressed videos using dynamic texture map together with artifact maps and 3D - fuzzy filters. To preserve better details during filtering process, the authors introduce a novel method to construct a texture map for video sequences called dynamic texture map. Then, temporal arifacts such as flicker artifacts and mosquito artifacts are also estimated by advanced flicker maps and mosquito maps. These maps combined with fuzzy filters are applied to intraframe and interframe pixels to enhance compressed videos. Simulation results verify the advanced performance of the proposed fuzzy filtering scheme in term of visual quality, SSIM, PSNR and flicker metrics in comparision with existing state of the art methods.**

Keywords– **Temporal artifact, dynamic texture, compression, fuzzy filter.**

1 Introduction

Video traffic is increasing dramatically fast. As in a study from Cisco [1], video traffic will achieve over 81% of the global traffic by year 2021. So, the requirement of compressing videos to reduce storage space and channel bandwidth is inevitable. There are many block-based compression standards such as JPEG, MPEG, H.26x, etc. to meet this requirement. However, these lossy compression methods suffer from spatial artifacts (blocking and ringing) and temporal artifacts (mosquito and flickering) ([2, 3]), especially at low bit rates. Blocking artifacts occur when the neightboring blocks are compressed independently. Beside that, the coarse quantization and truncation of high-frequency Discrete Cosine Transform (DCT) coefficients cause ringing artifacts. In interframe coding, at the borders of moving objects, the interframe predicted block may contain a part of the predicted moving object. The prediction error sometime is large and can cause mosquito artifacts. The authors in [4] and [5] introduce a method of flicker detection and reduction, however this method requires the original frames which are not available at the decoder.

Artifacts cause uncomfortableness to human visual perception. Hence, artifact removal becomes a very essential task. In general, image and video quality enhancement techniques can be implemented either at encoding side or decoding side. Enhancement methods at the encoding side ([6] and [7]) are not compatible to the existing video or image compression standards. Therefore, postprocessing techniques at the decoding side have received much more attention due to its compatibility to existing compression standards.

At the decoding side, Chen [8] proposes a method of image enhancement in the transform domain. In this method, the block DCT coefficients of shifted blocks are used to increase inter-block correlation. The filter window sizes adapt to the transform coefficients of classified blocks having low or high activities. In particular, a large window is utilized to efficiently smooth out blocking artifacts for low activity areas, while a smaller window is activated in high activity areas to keep the details. Furthermore, the authors in [9] introduce a blind measurement method of blocking artifacts. Based on the artifact level, an adaptive filter is then used to reduce blocking artifacts. In most multimedia post processings, video coding information is not available. In another research, the authors in [10] propose a quantization amount estimation method and then remove the ringing and mosquito artifacts in compressed video sequences by using spatio-temporal filtering. Recently, convolutional neural network (CNN) methods are applied to enhance image and video quality ([11] and [12]). CNNs are trained with original and compressed images and show to obtain rather good performance. However, the disadvantages of these methods is that it can create artificial details.

Enhancing image and video while preserving their details is very important. Thus the authors in [2], [13],

[14], [15], [16], and [17] use edge maps ([17] and [18]) to adapt filters' strength. However, the pixel classification is rather complicated and may lead to error. This can cause image details lost during filtering process. To better improve quality, accurate detection of image and video details is an unavoidable requirement. Image segmentation methods for locating texture are introduced in [19], [20], [21], and [22]. The authors in [23] use the enhanced Beltrami method to construct image texture maps. For the first time, the authors in [2] and [16] utilize the texture map together with fuzzy filters to remove blocking and ringing artifacts in compressed images. According to the best knowledge of the authors, texture maps for compressed video sequences still have not been studied.

In this paper, a novel method is proposed to enhance compressed video sequences. At first, a texture map of the compressed video sequence called the dynamic texture map is constructed. Then temporal artifact maps such as the flicker map and the mosquito map are estimated. The dynamic texture maps together with temporal artifact maps are used to control the fuzzy filters' strength. The remainings of this paper are organized as follows. Section 2 describes 3D-fuzzy filter. Section 3 constructs dynamic texture map. Section 4 introduces artifact maps of compressed video sequences. Video enhancement using dynamic texture and artifact maps is presented in Section 5. Simulation results and conclusions are shown in Section 6, and Section 7, respectively.

2 3D-Fuzzy Filters

3D-fuzzy filters are used to remove artifacts while preserving details of video frames. A 3D-fuzzy filter in [3] is applied to the input compressed video sequence I to formulate the output video sequence I' as

$$I'(x,y,t) = \frac{\sum\limits_{m,n,k\in\Omega} h(m,n,k) \times I\left(x+m, y+n, t+k\right)}{\sum\limits_{m,n,k\in\Omega} h(m,n,k)}, \tag{1}$$

where Ω are the neighbours of the pixel of interest $I(x,y,t)$; $h(m,n,k) = h(I(x,y,t), I(x+m,y+n,t+k))$ is the response function of the 3D-fuzzy filter. The filter response $h(I(x,y,t), I(x+m,y+n,t+k))$ must follow the constraints as in (2), (3), and (4)

$$\lim_{|I(x,y,t)-I(x+m,y+n,t+k)|\to 0} h(m,n,k) = 1, \tag{2}$$

$$\lim_{|I(x,y,t)-I(x+m,y+n,t+k)|\to +\infty} h(m,n,k) = 0, \tag{3}$$

and

$$\begin{aligned} &\text{if } |I(x,y,t) - I(x+m_1, y+n_1, t+k_1)| \\ &\qquad \geq |I(x,y,t) - I(x+m_2, y+n_2, t+k_2)|, \\ &h(I(x,y,t), I(x+m_1, y+n_1, t+k_1)) \\ &\qquad \leq h(I(x,y,t), I(x+m_2, y+n_2, t+k_2)). \end{aligned} \tag{4}$$

Gaussian function is one of the functions that satisfies the requirements in (2), (3), and (4)

$$\begin{aligned} &h(I(x,y,t), I(x+m,y+n,t+k)) \\ &\quad = \exp\left(-\frac{(I(x+m,y+n,t+k) - I(x,y,t))^2}{2\sigma^2(m,n,k)}\right), \end{aligned} \tag{5}$$

where the $\sigma(m,n,k)$ spread parameter of the Gaussian function adapts the strength of the 3D-fuzzy filters at different activity levels such as smooth or detail areas [2, 3].

3 Dynamic Texture Maps

Dynamic texture maps are constructed by classifying pixels based on their texture features. Pixels in the texture map are generally classified as strong edges, weak edges, strong texture, weak texture and flat areas. To be consistent to YUV decoded output during decompression, decompressed RGB video frames is first converted to YUV frames. At the decoding side, texture features of compressed video sequences are calculated as follows.

Let W be a cubic window of $(2M+1) \times (2N+1) \times (2K+1)$ pixels and $I(x,y,t)$ is the luminance value of the center pixel of W, where M, N, K are integer number. The texture map is constructed based on the Y component of pixels in video sequences. The Y component of pixels creates $\Re$ curve in the three-dimensional. Euclid distances of pixel values ([19]) in $\Re$ curve are calculated as in (6).

$$\begin{aligned} ds^2 &= dx^2 + dy^2 + dt^2 + dI^2 \\ &= dx^2 + dy^2 + dt^2 + \left(I_x dx + I_y dy + I_t dt\right)^2 \\ &= \left(1 + I_x^2\right) dx^2 + \left(1 + I_y^2\right) dy^2 + \left(1 + I_t^2\right) dt^2 \\ &\quad + 2I_x I_y dx dy + 2I_y I_t dy dt + 2I_t I_x dt dx. \end{aligned} \tag{6}$$

The feature matrix is defined as in (7)

$$g_{xyt} = \begin{bmatrix} 1 + I_x^2 & I_x I_y & I_x I_t \\ I_y I_x & 1 + I_y^2 & I_y I_t \\ I_t I_x & I_t I_y & 1 + I_t^2 \end{bmatrix}. \tag{7}$$

Let I_x, I_y, I_t be partial derivatives along x, y, and t directions in the cubic window W. These derivatives are calculated as in (8), (9) and (10)

$$I_x = \sqrt{\frac{\sum\limits_{m=-M}^{M}\sum\limits_{n=-N}^{N}\sum\limits_{k=-K}^{K}\left[W\left(m+1,n,k\right) - W\left(m,n,k\right)\right]^2}{(2M+1)\times(2N+1)\times(2K+1)}} \tag{8}$$

$$I_y = \sqrt{\frac{\sum\limits_{m=-M}^{M}\sum\limits_{n=-N}^{N}\sum\limits_{k=-K}^{K}\left[W\left(m,n+1,k\right) - W\left(m,n,k\right)\right]^2}{(2M+1)\times(2N+1)\times(2K+1)}} \tag{9}$$

$$I_t = \sqrt{\frac{\sum\limits_{m=-M}^{M}\sum\limits_{n=-N}^{N}\sum\limits_{k=-K}^{K}\left[W\left(m,n,k+1\right) - W\left(m,n,k\right)\right]^2}{(2M+1)\times(2N+1)\times(2K+1)}} \tag{10}$$

The texture feature of video sequences is defined

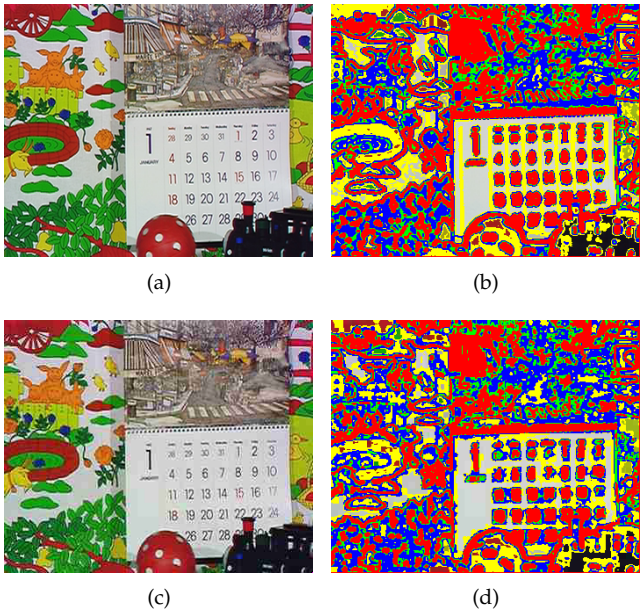
in [17] as in(11)

$$F(x,y,t) = \exp\left(-\frac{\det\left(g_{xyt}\right)}{\delta^2}\right), \tag{11}$$

where δ is a scaling parameter. Pixel classification is determined based on the texture feature value as in (12). Classification thresholds is experientially selected and dynamic texture map quality bases on subjective assessment of observers. These thresholds determine the accuracy of the dynamic texture map, so they influence to preserve details of video frames when video enhancement is implemented by the 3D-fuzzy filters.

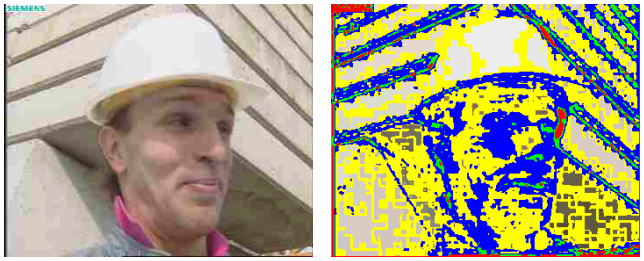
$$\begin{aligned} &Pel\text{-}type = \\ &= \begin{cases} Strong-edge & F(x,y,t) < 10^{-4} \\ Weak-edge & 10^{-4} \le F(x,y,t) < 8.10^{-3} \\ Strong-texture & 8.10^{-3} \le F(x,y,t) < 0.5 \\ Weak-texture & 0.5 \le F(x,y,t) < 0.95 \\ Flat & otherwise. \end{cases} \end{aligned} \tag{12}$$

Figure 1 shows an example of the dynamic texture map of Mobile video sequence. Colors of the texture map are defined as follows. Red: strong edge; Green: weak edge; Blue: strong texture; Yellow: weak texture; Others: flat. Figure 1(a), Figure 1(b), Figure 1(c), and Figure 1(d), respectively are the original frame, the dynamic texture map of the original frame, the MJPEG compressed frame, and the dynamic texture map of the MJPEG compressed frame. Visually, the dynamic texture maps show acurately the details of video frames in classifying strong edge, weak edge, strong texture, weak texture, and flat areas. As can seen in Figure 1(b) and Figure 1(d), the dynamic texture map of original frame are more accurate details than that of compressed frame. Figure 2 and Figure 3 show the MJPEG compressed frames and their texture maps.



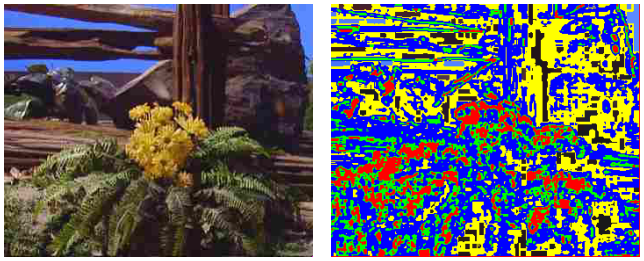
(a) (b) (c) (d)

Figure 1. An example of the dynamic texture map. (a) Original frame; (b) the dynamic texture map of the original frame; (c) the MJPEG compressed frame; (d) the dynamic texture map of the MJPEG compressed frame.



(a) (b)

Figure 2. The 5^{th} frame of Foreman video sequence. (a) the MJPEG compressed frame; (b) the dynamic texture map.



(a) (b)

Figure 3. The 5^{th} frame of Tempete video sequence. (a) the MJPEG compressed frame; (b) the dynamic texture map.

4 Artifact Maps of Compressed Video Sequences

4.1 Flicker Maps

Flicker artifact is a temporal artifact that causes annoyance feeling of viewers. Flicker artifacts occur as the sudden luminance value changes of the co-located pixels between two consecutive frames. This subsection introduces a novel method of constructing a flicker map. The original video frames usually are not available at the decoding side, so this paper proposes to use compressed video frames to estimate flicker maps. This method compares luminance difference of the co-located pixels in neighbour compressed video frames to determine flickering pixels.

The proposed method only uses the Y component of the pixels to construct the flicker maps. Let $I(x,y,t)$ be luminance value at position (x,y) in t^{th} frame, $I^{'}(x,y,t-1)$ be the corresponding reconstructed pixel in previous frame ($x = 1,\cdots,H; y = 1,\cdots,D$, where H and D respectively are the height and width of the video frame). The authors define the luminance difference of two co-located pixels between two consecutive frames as in (13).

$$\Delta^{\tau}_{x,y} = \left| I(x,y,t) - I^{'}(x,y,t-1) \right| \tag{13}$$

Assume that a number of frames consider to be an even integer $2n$, $\tau = 1,\cdots,2n-1$.

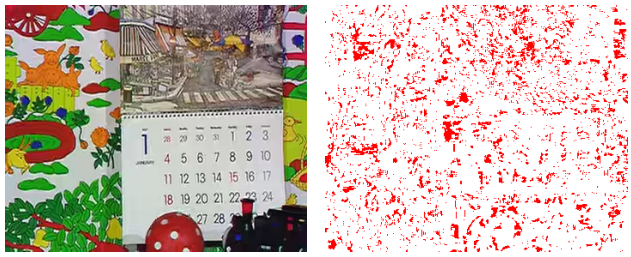
Let $\tau_c = \lfloor \frac{\tau}{2} \rfloor$. If $\max\left(\Delta^{\tau}_{x,y}\right) < \Delta^{\tau_c}_{x,y}$ $(\tau \neq \tau_c)$ then $I(x,y,\tau_c)$ is the gap pixel and considered as flicker artifact pixel. Therefore, based on neighbour frames, the

$\Delta^1_{x,y}$	$\Delta^2_{x,y}$	$\Delta^3_{x,y}$	$\Delta^4_{x,y}$	$\Delta^5_{x,y}$

Figure 4. Different luminance values of the co-located pixels.

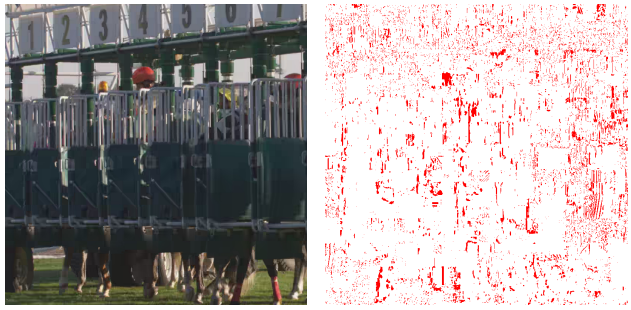
(a) (b)

Figure 5. Flicker map of the 5^{th} frame in Foreman video sequence. (a) Coded frame; (b) Flicker map.



(a) (b)

Figure 6. Flicker map of the 5^{th} frame in Mobile video sequence. (a) Coded frame; (b) Flicker map.

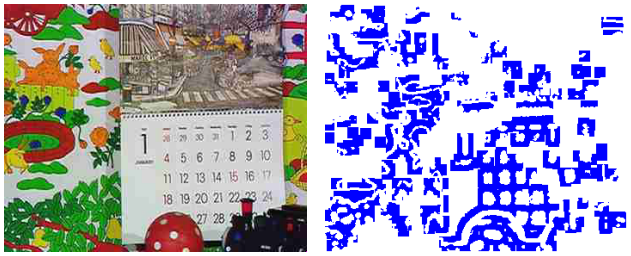


(a) (b)

Figure 7. Flicker map of the 12^{th} frame in the zoomed ReadySteadyGo video sequence. (a) Coded frame; (b) Flicker map.

t^{τ_c} frame is used to construct the flicker map. Similarly, process is repeated to estimate the flicker maps for a whole compressed video sequence. Figure 4 is an example of the proposed method when constructing a flicker map with neighbour frames of $2n = 6$.

Figure 5 shows an example of flicker map of the 5^{th} frame in Foreman video sequence, where red color is flicker artifact pixels. Figure 5(a) and Figure 5(b), respectively are the coded frame and the flicker map. Similarly, Figure 6 and Figure 7 are the flicker maps in Mobile and ReadySteadyGo video sequences. As can be seen in these results, flickering pixels appear many at borders of edges due to the quantization errors and the predictive errors of blocks during encoding video sequences.



(a) (b)

Figure 8. Mosquito map of the 5^{th} frame in Mobile video sequence. (a) Coded frame; (b) Mosquito map.

(a) (b)

Figure 9. Mosquito map of the 5^{th} frame in YachtRide video sequence. (a) Coded frame; (b) Mosquito map.

4.2 Mosquito Maps

Predictive errors of blocks along the border of moving objects create mosquito artifacts. So, neighbour pixels of moving object borders are considered as potential mosquito artifacts. To determine moving object borders, the authors define a border block content that contains at least one pixel of the border edges. If there is a relative motion between a border block and its neighbour blocks, it is considered as a border block of moving objects.

Let MV_c be a motion vector of the border block, MV^l_{nb} be the motion vectors of the neighbour blocks of the border block ($l = 1, \cdots, 8$). The distance between the border block and its l^{th} neighbour block is calculated as in (14)

$$d_l = \left\| MV^l_{nb} - MV_c \right\|_2, \tag{14}$$

where $\|.\|_2$ is norm-2. If any d_l value is greater or equal to 1 then it is considered as the border block of moving objects. So, the neighbour pixels of this border block but the moving object pixels are estimated the mosquito pixels. Figure 8 shows an example of mosquito map of the 5^{th} frame in Mobile video sequence, where blue color is mosquito artifact pixels. Figure 8(a) and Figure 8(b), respectively are coded frame and mosquito map. Similarly, Figure 9 is the mosquito map of the 5^{th} frame in YachtRide video sequence. As can be seen in these results, potential mosquito pixels follow at the moving object borders. That shows the mosquito map of the proposed method is suitable for characteristic of the mosquito artifact pixels.

5 Video Enhancement using Dynamic Texture Maps

Compressed video enhancement is the target of this paper. The authors propose a novel method to combine

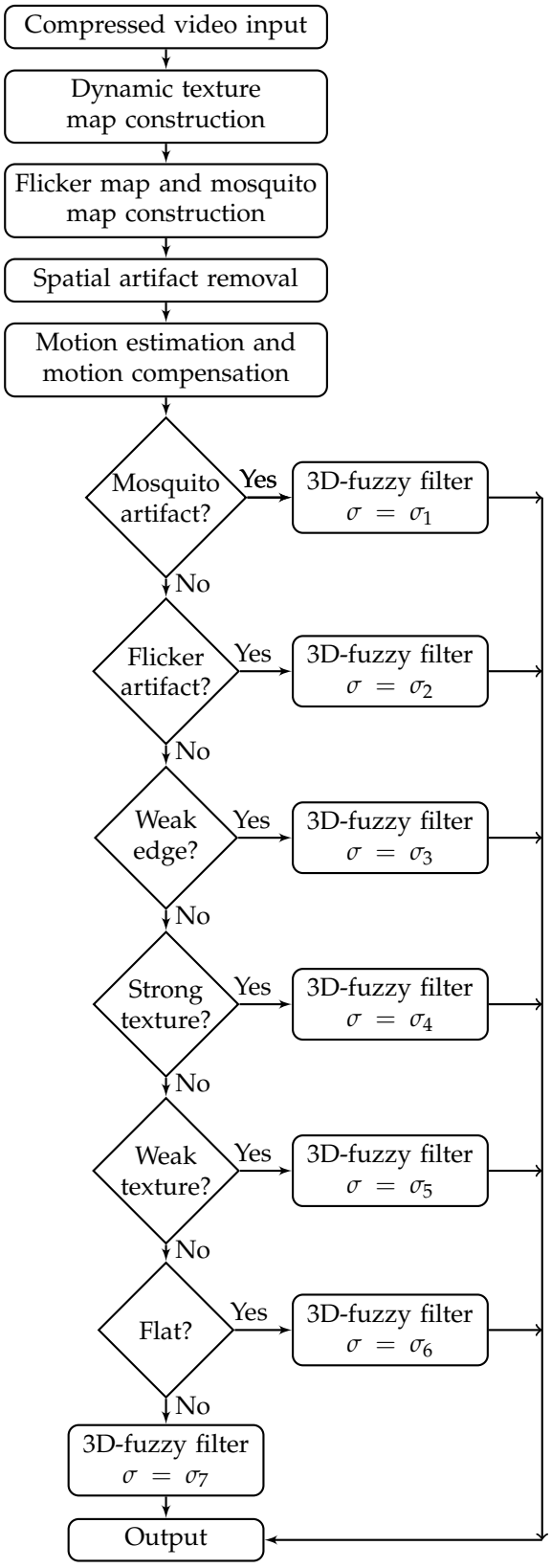


Figure 10. The flow chart of the proposed video enhancement method.

dynamic texture maps, temporal artifact maps, and 3D-fuzzy filters to remove artifacts. Dynamic texture maps is used to control strength of 3D-fuzzy filters at every pixel in compressed video sequences. Figure 10 shows the flow chart of the proposed video enhancement method. The input of this scheme is a compressed video sequence, which suffers from spatial artifacts and temporal artifacts. Video enhancement implements both spatial and temporal domains. Spatial filtering removes blocking and ringing artifacts in every frame. Temporal filtering removes flicker and mosquito artifacts in coded videos. The authors in [2] propose methods of removing spatial artifacts based on texture maps, spatial artifact maps, and 2D-fuzzy filters. However, spatial artifact removal may smooth out and distort the details in compressed video frames. Estimation of the dynamic texture map, flicker map, and mosquito map based on the video after spatial artifact enhancement is able to cause exactly missing. Therefore, the paper proposes to construct these maps based on coded videos. As mentioned in Section 3 and Section 4, dynamic texture maps, flicker maps, and mosquito maps are constructed from compressed video input. In this paper, motion vector estimation and motion compensation is implemented to further increase the correlation of neighbour pixels. Pixels are classified into flicker artifact, mosquito artifact, strong edge, weak edge, strong texture, weak texture, and flat. The strength of 3D-fuzzy filters corresponding with types of pixels in filtering process are controlled by the spread parameters $\{\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6, \sigma_7\}$. These parameters are experientially selected to adapt filters' strength based on the dynamic texture map. If the pixel is a mosquito pixel, it is filtered by a 3D-fuzzy filter with σ_1 value to remove the mosquito artifacts. Else if the pixel is a flicker pixel, it is filtered by a 3D-fuzzy filter with σ_2 value to remove the flicker artifacts. Next, depending on the texture types (which are weak edges, strong texture, weak texture, and flat), corresponding the 3D-fuzzy filters with different spread parameters are applied to remove artifacts while preserving details of video frames. The spread parameter values are ranged from the highest values σ_1 and σ_2 for mosquito and flicker areas to the lowest value σ_7 for flat areas, corresponding to the strongest filtering level to the weakest filtering level.

To adapt to different areas having different activity levels, the amplitude of the spread parameter to control the strength of the 3D-fuzzy filters is defined ([2]) as in (15)

$$\sigma(x,y,t) = \sigma_i \left((1-\gamma) \left(\frac{F_{\max} - F(x,y,t)}{F_{\max} - F_{\min}} \right) + \gamma \right), \tag{15}$$

where $F_{\min}$ and $F_{\max}$ are minimum and maximum values of all $F(x,y,t)$ values defined as in (11), γ is a scaling factor in $[0,1]$, and σ_i $(i = 1,...,7)$ is the selected spread parameter value. The parameters in [2] are selected as α=0.5, β=3.5 and γ=0.5.

6 Simulation Results

To validate the effectiveness of the proposed method, MATLAB programs are implemented to stimulate the results on a computer with Intel(R)Core(TM)i7-8550U CPU@1.8 GHz up to 4 GHz and RAM of 8 GB.

Table I
PSNR Comparison of the Proposed Method for MJPEG Video Sequences

Sequences	MJPEG	Chen	Liu	MCSTF	CNN	Proposed
Highway	31.5952	31.7564	31.6935	32.3826	31.3481	32.5535
Silent	29.5057	30.0548	29.8812	29.8099	28.6818	30.4979
Mother	32.8453	33.4530	33.2049	33.4878	33.7641	34.1748
Foreman	29.6542	30.0627	29.9964	30.6263	29.1103	31.0253
Mobile	23.2985	23.1300	23.2383	23.4489	24.1733	24.0169
Bridge-far	31.2002	31.3582	31.3883	31.7274	29.3216	31.9052
Tempete	25.4873	25.7231	25.6077	25.4210	26.1018	26.1642
Bridge-close	27.8246	27.8812	27.8964	27.6414	24.7826	28.4391
Hall	30.4926	30.2846	30.3200	30.2465	31.4891	31.3879
Bosphorus	34.3488	35.0894	34.7790	35.0636	34.8794	35.5651
HoneyBee	32.4267	33.1781	33.1987	33.9100	32.8354	33.9018
ReadySteadyGo	31.5391	32.1103	31.7610	31.7031	32.3285	32.3676
Averaged difference		0.3220	0.2289	0.4375	-0.1169	0.9818

Table II
SSIM Comparison of the Proposed Method for MJPEG Video Sequences

Sequences	MJPEG	Chen	Liu	MCSTF	CNN	Proposed
Highway	0.8259	0.8483	0.8385	0.8542	0.8576	0.8647
Silent	0.8176	0.8267	0.8186	0.7871	0.8372	0.8384
Mother	0.8697	0.8872	0.8776	0.8679	0.8985	0.8960
Foreman	0.8130	0.8363	0.8285	0.8372	0.8516	0.8509
Mobile	0.8302	0.8216	0.8263	0.8126	0.8774	0.8630
Bridge-far	0.7488	0.7714	0.7738	0.7775	0.7817	0.7799
Tempete	0.8400	0.8412	0.8381	0.8015	0.8717	0.8598
Bridge-close	0.7702	0.7657	0.7648	0.7087	0.7786	0.7822
Hall	0.8550	0.8677	0.8632	0.8691	0.8937	0.8876
Bosphorus	0.8866	0.9048	0.8915	0.8756	0.9029	0.9027
HoneyBee	0.8862	0.9127	0.9058	0.9174	0.9208	0.9214
ReadySteadyGo	0.8955	0.9113	0.9000	0.8863	0.9208	0.9049
Averaged difference		0.0130	0.0073	-0.0036	0.0295	0.0261

Table III
Flicker Comparison of the Proposed Method for MJPEG Video Sequences

Sequences	MJPEG	Chen	Liu	MCSTF	CNN	Proposed
Highway	3.2291	3.2477	3.5684	3.188	3.9039	2.9907
Silent	7.0942	6.2600	6.6188	5.9567	7.9481	5.6088
Mother	0.5615	0.5874	0.5615	0.3957	0.8751	0.2623
Foreman	2.2877	2.1748	2.2930	2.1701	2.7972	1.8121
Mobile	3.1923	2.5145	3.2470	2.2008	3.0447	1.5207
Bridge-far	1.0169	1.0065	1.0546	0.9082	1.4470	0.8530
Tempete	2.1304	1.7345	2.1094	1.4376	2.2063	1.3085
Bridge-close	2.3861	2.1964	2.4817	2.2325	3.1765	1.7577
Hall	3.4817	3.4935	4.0280	3.0824	3.6551	2.9436
Bosphorus	0.2055	0.1835	0.2032	0.1665	0.2040	0.1495
HoneyBee	0.4621	0.3593	0.3748	0.2905	0.3513	0.2973
ReadySteadyGo	1.5646	1.2718	1.3365	1.0556	1.3355	1.0287
Averaged difference		-0.2152	0.0221	-0.3773	0.2777	-0.5899

Objective and subjective assessments based on PSNR, SSIM [21], the flicker metric [3], and the visual quality are used to compare video enhancement methods. Averaged value of each quality index in successive frames are also calculated. Many original video sequences with different resolutions are compressed with MJPEG and H.265 standards, and then enhanced by many different methods. Chosen video sequences with resolutions of 352×288 (Highway, Silent, Mother, Foreman, Mobile, Bridge-far, Tempete, Bridge-close, and Hall) and 1920×1080 (Bosphorus, HoneyBee, ReadySteadyGo, YachtRide, and Beauty) are used to stimulate and compare the results of enhacement methods. Spread parameters of 3D-fuzzy filters in Figure 10 are experimently selected as σ_1 =16, σ_2 =16, σ_3 =12, σ_4=11, σ_5 =11, σ_6 =10, σ_7 =8, and σ_8 =12. Filter window of the 3D-fuzzy filter in this paper is proposed to be 3×3×5. Details are shown in two subsection as follows.

6.1 Enhancement for MJPEG Encoded Video Sequences

The original video frames are compressed with MJPEG standard, which do not consider temporal correlation. Table I, Table II, and Table III, respectively show the comparision in PSNR values, SSIM values, and the flicker metric values among the proposed method, Chen [8], Liu [9], MCSTF [3], and CNN [11]. The averaged PSNR improvement values of the Chen method, the Liu method, the MCSTF method, the CNN method, and the proposed method over the MJPEG encoded video sequences are +0.3220 dB, +0.2289 dB, +0.4375 dB, -0.1169 dB, and 0.9818 dB, respectively. The averaged SSIM improvement values of the Chen method, the Liu method, the MCSTF method, the CNN method, and the proposed method are +0.0130, +0.0073, -0.0036, +0.0295, and +0.0261, respectively. The averaged flicker metric improvement values of the Chen method, the Liu method, the MCSTF method, the CNN method, and the proposed method are -0.2152, +0.0221, -0.3773, -0.2777, and -0.5899, respectively.

In comparison with the existing methods such as the Chen method, the Liu method, the MCSTF method, and the CNN method, the averaged PSNR improvement value of the proposed method is +0.6598 dB, +0.7528 dB, +0.5442 dB, and +1.0986 dB, respectively; similarly, the averaged SSIM improvement value of the proposed method is +0.0131, +0.0187, +0.0297, and -0.0034; and the averaged flicker metric improvement value of the proposed method is -0.3748, -0.6120, -0.2126, and -0.8677. The averaged PSNR improvement value of the CNN method and the averaged SSIM improvement value of the MCSTF method are smaller than those of MJPEG encoded sequences. The averaged flicker metric improvement value of the CNN method and the Liu method are larger than that of MJPEG encoded se-

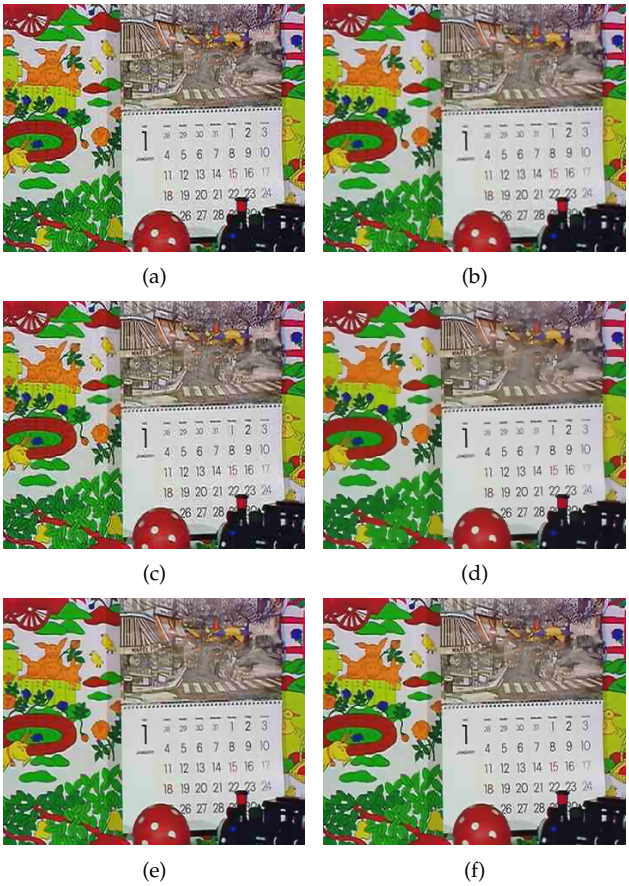


Figure 11. The 10^{th} frame of Mobile video sequence. (a) Compressed; (b) Chen; (c) Liu; (d) MCSTF; (e) CNN; (f) The proposed method.

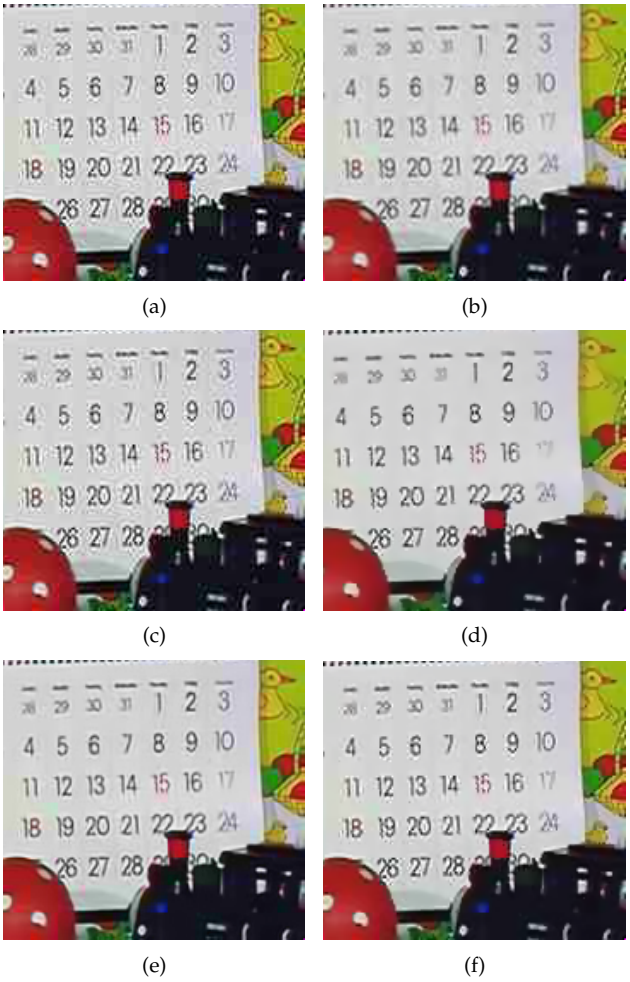


Figure 12. The zoomed 10^{th} frame of Mobile video sequence. (a) Compressed; (b) Chen; (c) Liu; (d) MCSTF; (e) CNN; (f) The proposed method.

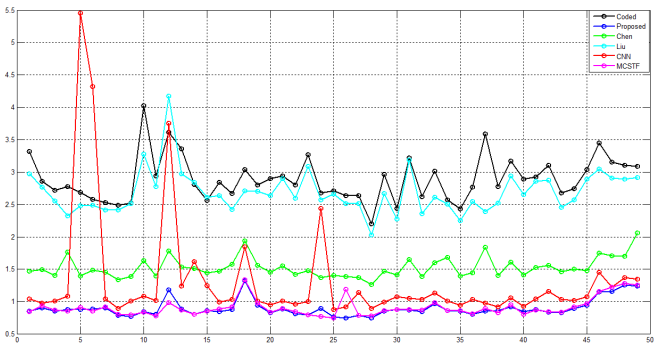


Figure 13. Flicker metric comparision in Mobile video sequence.

quences. This means these methods do not improve the flicker metric of MJPEG encoded sequences. PSNRs and the flicker metrics of the proposed method is improved significantly more than those of other methods, SSIMs of the proposed method is equivalent to those of the CNN method and is improved better than those of the other methods. Video enhancement of Chen, Liu, and CNN methods only implements in single frame based mode (called intraframe mode), so the flicker metric of these methods is not consistent over frames. The MCSTF method and the proposed method are performed in single frame based mode and multi frame based mode (called interframe mode) to remove both spatial and temporal artifacts. Therefore, the flicker metric of the MCSTF method and the proposed method are much better than the other methods.

To evaluate the visual quality, the enhancement results of the different methods on the 10^{th} frame of the Mobile video sequence are shown on Figure 11 and Figure 12. As can be seen in these results, Chen method result (Figure 12(b)) is blurry; the Liu method result (Figure 12(c)) still has many ringing artifacts, the MCSTF method introduces good result but many details are lost (Figure 12(d)); the CNN method result (Figure 12(e)) and the proposed method result (Figure 12(f)) improve quality better than the other methods. However, according to Figure 13, the flicker metric of the CNN method is not improved. Similarly, the results of the 10^{th} frame of the Foreman video sequence are shown on Figure 14 and Figure 15, where the proposed method, the MCSTF method, and the CNN method introduce the best qualities. However, the result of the CNN method is color bleeding, the result of the MCSTF method is lost many details.

In the MJPEG encoded video sequcence enhancement, based on the above results, the proposed method outperforms the other methods in term of PSNRs, SSIMs (except the CNN method), the flicker metrics, and the visual quality. The SSIM value of the CNN method and the proposed method are equavalent each other.

Figure 14. The 10^{th} frame of Foreman video sequence. (a) Compressed; (b) Chen; (c) Liu; (d) MCSTF; (e) CNN; (f) The proposed method.

Table IV
PSNR Comparison of the Proposed Method for H.265 Video Sequences

Sequences	H.265	Chen	Liu	MCSTF	CNN	Proposed
Highway	31.6271	30.9281	31.3742	31.8720	30.8110	31.8736
Mother	32.5383	32.6104	32.4123	32.4226	32.7943	32.7198
Foreman	30.0859	30.7423	30.4011	31.2270	29.2476	31.4596
Mobile	26.0222	23.1958	25.2848	24.9134	25.6546	25.8499
Bridge-far	31.5543	31.3683	31.4304	31.6530	29.4862	31.7308
Bridge-close	27.5614	27.2508	27.4861	27.2326	24.4270	27.6360
Hall	29.8270	29.1281	29.3884	29.1417	30.0891	29.7552
Beauty	33.3822	33.4866	33.4464	33.5123	33.5496	33.5153
Bosphorus	36.2591	36.1707	35.6105	35.4355	36.0618	36.0361
Honey	32.8902	33.1216	33.0254	33.2205	33.1163	33.3114
ReadySteadyGo	30.8218	30.8305	30.5702	30.5857	31.2092	30.9901
YachtRide	32.0129	32.1024	31.8633	31.9799	32.3608	32.1844
Averaged difference		-0.3039	-0.1908	-0.1155	-0.4812	0.2067

6.2 Enhancement for H.265 Encoded Video Sequences

H.265 standard is the lastest video encoding standard. In this subsection, the original frames are compressed using this standard. The authors simulate different methods to enhance the H.265 video sequences. The configuration parameters encoding the H.265 standard are as follows: the prediction structure is IPP-PIPPP, QP (Quality Parameter) is 38, and deblocking and deringing filters are turn off. The objective simulation results of the Chen method, the Liu method, the MCSTF method, the CNN method, and the proposed method are shown in Table IV, Table V, and Table VI, respectively. The averaged PSNR improvement values of the Chen method, the Liu method, the MCSTF method, the CNN method, and the proposed method are -0.3039 dB, -0.1908 dB, -0.1155, -0.4812 dB, and +0.2067 dB, respectively. In comparison of the PSNR value of the H.265 encoded video sequences, only the proposed method provides improvement while the other methods do not. The averaged SSIM improvement values of the Chen method, the Liu method, the MCSTF method, and the proposed method are -0.0001, -0.0027,

Figure 15. The zoomed 10^{th} frame of Foreman video sequence. (a) Compressed; (b) Chen; (c) Liu; (d) MCSTF; (e) CNN; (f) The proposed method.

Table V
SSIM Comparison of the Proposed Method for H.265 Video Sequences

Sequences	H.265	Chen	Liu	MCSTF	CNN	Proposed
Highway	0.8064	0.8085	0.8039	0.8237	0.8185	0.8233
Mother	0.8357	0.8379	0.8323	0.8285	0.8427	0.8394
Foreman	0.7741	0.8431	0.7809	0.8462	0.8577	0.8579
Mobile	0.8943	0.8097	0.8842	0.8426	0.8821	0.8842
Bridge-far	0.7522	0.7543	0.7544	0.7591	0.7577	0.7591
Bridge-close	0.6943	0.6791	0.6860	0.6531	0.6861	0.6848
Hall	0.8451	0.8412	0.8423	0.8428	0.8624	0.8520
Beauty	0.7517	0.7574	0.7545	0.7571	0.7605	0.7571
Bosphorus	0.8975	0.8953	0.8802	0.8607	0.8816	0.8847
Honey	0.8716	0.8836	0.8786	0.8885	0.8894	0.8872
ReadySteadyGo	0.8575	0.8617	0.8518	0.8496	0.8690	0.8629
YachtRide	0.8254	0.8332	0.8244	0.8300	0.8404	0.8353
Averaged difference		-0.0001	-0.0027	-0.0020	0.0119	0.0102

Table VI
Flicker Comparison of the Proposed Method for H.265 Video Sequences

Sequences	H.265	Chen	Liu	MCSTF	CNN	Proposed
Highway	3.2593	3.1835	3.2360	3.1025	3.6555	3.0008
Mother	0.2731	0.2370	0.2731	0.3217	0.2888	0.3310
Foreman	2.4169	1.5799	2.3516	1.3126	1.9443	1.1598
Mobile	1.3999	1.2856	1.3952	1.0346	1.2024	0.9828
Bridge-far	0.8340	0.8606	0.8546	0.7991	1.1422	0.7954
Bridge-close	1.7015	2.0159	2.0716	1.9880	2.0140	1.7613
Hall	2.9285	3.1165	3.2249	3.0004	2.8316	2.8899
Beauty	0.8486	0.8126	0.8223	0.7271	0.7928	0.7418
Bosphorus	0.2127	0.1951	0.2025	0.1554	0.2005	0.1472
Honey	0.4196	0.3368	0.3380	0.3061	0.3133	0.3070
ReadySteadyGo	1.4680	1.2576	1.3310	1.1425	1.1787	1.1530
YachtRide	0.3846	0.3564	0.3500	0.3115	0.3130	0.3061
Averaged difference		-0.0758	0.0253	-0.1621	-0.0225	-0.2142

-0.0020, +0.0119, and +0.0102, respectively. In comparison of the SSIM value of the H.265 encoded video sequences, the proposed method is slightly higher and is in similar level with the CNN method while the other methods are lightly lower. The averaged flicker metric improvement values of the Chen method, the Liu method, the MCSTF method, and the proposed method are -0.0758, +0.0253, -0.1621, -0.0225, and -0.2142, respectively. In comparison of the Chen method, the Liu method, the MCSTF method, and the CNN method, the averaged PSNR improvement value of the proposed method increases +0.5155, +0.3974, +0.3222, and +0.6879, respectively. Similarly, the averaged SSIM improvement value of the proposed method increases +0.0102, +0.0129, +0.0122, and -0.0017. The averaged flicker metric improvement value of the proposed method decreases -0.1385, -0.2396, -0.0521, and -0.1918. The H.265 standard is benefical from both intraframe and interframe encoding structure to enhance flicker metric and optimise PSNR value. So, in H.265 video sequences, PSNRs of the proposed method improve less than in MJPEG video sequences although PSNRs of the proposed method still are the best among compared methods. Flicker metric of the proposed method also improves better than that of the other methods. The averaged SSIM improvement value of the propose method is equivalent to that of the CNN method and higher than the other methods.

7 Conclusions

Lossy compression introduces annoying artifacts to visual human. This paper proposes a novel method to enhance the compressed video sequences. With the compressed video sequence input, the authors implement advanced methods to construct the dynamic texture map, the flicker artifact map and the mosquito artifact map. These maps are used to control the fuzzy filter's strength to remove artifacts while significantly preserving more details of the compressed video sequences. The simulation results show that the proposed method improves effectively in terms of PSNR, SSIM, flicker metric and visual quality in comparision with other state of the art methods.

References

[1] Cisco Systems, "Cisco visual networking index: Forecast and trends, 2017–2022," *White Paper*, 2018. [Online]. Available: https://www.cisco.com/c/en/ us/ solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html

[2] T. Van Nguyen, T. H. Do, and D. T. Vo, "A novel joint blocking and ringing artifact reduction using advanced beltrami based texture maps," *REV Journal on Electronics and Communications*, vol. 7, no. 1-2, 2017.

[3] D. T. Vo, T. Q. Nguyen, S. Yea, and A. Vetro, "Adaptive fuzzy filtering for artifact reduction in compressed images and videos." *IEEE Transactions on Image Processing*, vol. 18, no. 6, pp. 1166–1178, 2009.

[4] A. Jiménez-Moreno, E. Martínez-Enríquez, and F. Díaz-de María, "Standard compliant flicker reduction method with PSNR loss control," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 1729–1733.

[5] A. Jiménez-Moreno, E. Martinez-Enriquez, V. Kumar, and F. Díaz-de María, "Standard-compliant low-pass temporal filter to reduce the perceived flicker artifact," *IEEE Transactions on Multimedia*, vol. 16, no. 7, pp. 1863–1873, 2014.

[6] M. Kaneko, Y. Hatori, and A. Koike, "Improvements of transform coding algorithm for motion-compensated interframe prediction errors-dct/sq coding," *IEEE Journal on Selected Areas in Communications*, vol. 5, no. 7, pp. 1068–1078, 1987.

[7] H. S. Malvar and D. H. Staelin, "The lot: Transform coding without blocking effects," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 4, pp. 553–559, 1989.

[8] T. Chen, H. R. Wu, and B. Qiu, "Adaptive postfiltering of transform coefficients for the reduction of blocking artifacts," *IEEE Transactions on Circuits and Systems For Video Technology*, vol. 11, no. 5, pp. 594–602, 2001.

[9] S. Liu and A. C. Bovik, "Efficient DCT-domain blind measurement and reduction of blocking artifacts," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 12, pp. 1139–1149, 2002.
[10] S. B. Yoo, K. Choi, and J. B. Ra, "Blind post-processing for ringing and mosquito artifact reduction in coded videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 5, pp. 721–732, 2013.
[11] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2015.
[12] K. Yu, C. Dong, C. C. Loy, and X. Tang, "Deep convolution networks for compression artifacts reduction," in *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, Feb. 2016.
[13] H.-S. Kong, Y. Nie, A. Vetro, H. Sun, and K. E. Barner, "Adaptive fuzzy post-filtering for highly compressed video," in *Proceedings of the International Conference on Image Processing (ICIP'04)*, vol. 3. IEEE, Oct. 2004, pp. 1803–1806.
[14] H.-S. Kong, A. Vetro, and H. Sun, "Edge map guided adaptive post-filter for blocking and ringing artifacts removal," in *Proceedings of the International Symposium on Circuits and Systems (IEEE Cat. No. 04CH37512)*, vol. 3. IEEE, May 2004, pp. III–929.
[15] E. Nadernejad, S. Forchhammer, and J. Korhonen, "Artifact reduction of compressed images and video combining adaptive fuzzy filtering and directional anisotropic diffusion," in *Proceedings of the 3rd European Workshop on Visual Information Processing*. IEEE, 2011, pp. 24–29.
[16] T. Van Nguyen, T. H. Do, and D. T. Vo, "Advanced texture-adapted blocking removal for compressed visual content," in *Proceedings of the International Conference on Advanced Technologies for Communications (ATC)*. IEEE, Oct. 2015, pp. 285–290.
[17] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, "Design of an image edge detection filter using the sobel operator," *IEEE Journal of Solid-State Circuits*, vol. 23, no. 2, pp. 358–367, 1988.
[18] A. Jain, M. Gupta, S. Tazi, and Deepika, "Comparison of edge detectors," in *Proceedings of the International Conference on Medical Imaging, m-Health and Emerging Communication Systems (MedCom)*. IEEE, 2014, pp. 289–294.
[19] N. Sochen, R. Kimmel, and R. Malladi, "A general framework for low level vision," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 310–318, 1998.
[20] N. Houhou, J.-P. Thiran, and X. Bresson, "Fast texture segmentation based on semi-local region descriptor and active contour," *Numerical Mathematics: Theory, Methods and Applications*, vol. 2, no. 4, pp. 445–468, 2009.
[21] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. IEEE, 1999, pp. 1033–1038.
[22] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and H.-Y. Shum, "Real-time texture synthesis by patch-based sampling," *ACM Transactions on Graphics (ToG)*, vol. 20, no. 3, pp. 127–150, 2001.
[23] T. Van Nguyen, D. T. Vo, and T. H. Do, "Highly noise resistant beltrami-based texture maps with window derivative," in *Proceedings of the 2nd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS)*. IEEE, 2015, pp. 71–75.

Thai Van Nguyen graduates in telecommunication electronics at the Can Tho University in 2003 and receives M.S degree in electronics engineering from the Ho Chi Minh City University of Technology (HCMUT), HCMC, Vietnam, in 2009. He is currently pursuing the Ph.D. degree at HCMUT. He is working at MobiFone Corporation, Vietnam. His research interests are image and video quality enhancement, MIMO.

Tuan Do-Hong received the B.S. and M. Eng. degrees in electrical engineering from the Ho Chi Minh City University of Technology, Vietnam National University Ho Chi Minh city, Vietnam, in 1994 and 1997, respectively, the M.Sc. and Ph.D. degrees in communication engineering from the Munich University of Technology, Germany, in 2000 and 2004, respectively. He has been is Dean of Faculty of Electrical and Electronics Engineering, the Ho Chi Minh City University of Technology, Vietnam National University Ho Chi Minh City, Vietnam. His research interests include stochastic signal processing and applications for image and video processing.

Dung Trung Vo (S'06 - M'09) received the B.S. and M.S. degrees from HCMUT, Vietnam, in 2002 and 2004, respectively, and the Ph.D. degree from the University of California at San Diego, La Jolla, in 2009. He has been a Fellow of the Vietnam Education Foundation (VEF) and is with HCMUT since 2002. He interned at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, and Thomson Corporate Research, Princeton, NJ, in the summers of 2007 and 2008, respectively. He has been a staff 2 research engineer at the Digital Media Solutions Lab, Samsung Research America, Irvine, CA, since 2009. He receives the Special Merit Awards for Outstanding Paper at IEEE Conference on Consumer Electronics (ICCE) 2011 and 2012. His research interests are algorithms and applications for image and video coding and post-processing.