Improving Non-Local Video Denoising with Local Binary Patterns and Image Quantization

Welinton A. Contato, Tiago S. Nazare, Gabriel B. Paranhos da Costa, Moacir Ponti, João E.S. Batista Neto
Instituto de Ciências Matemáticas e de Computação (ICMC) – Universidade de São Paulo (USP)
São Carlos/SP – 13566-590, Brazil
Email: {welintonandrey, tiagosn, gbpcosta, ponti}@usp.br, jbatista@icmc.usp.br

***Abstract*—The most challenging aspect of video and image denoising is to preserve texture and small details, while filtering out noise. To tackle such problem, we present two novel variants of the 3D Non-Local Means (NLM3D) which are suitable for videos and 3D images. The first proposed algorithm computes texture patterns for each pixel by using the LBP-TOP descriptor to modify the NLM3D weighting function. It also uses MSB (Most Significant Bits) quantization to improve robustness to noise. The second proposed algorithm filters homogeneous and textured regions differently. It analyses the percentage of non-uniform LBP patterns of a region to determine whether or not the region exhibits textures and/or small details. Quantitative and qualitative experiments indicate that the proposed approaches outperform well known methods for the video denoising task, especially in the presence of textures and small details.**

***Keywords*-local binary patterns; most significant bits; non-local means; video denoising;**

I. Introduction

Despite recent improvements on camera sensors, image and video denoising still is an important research area as the amount of observed noise may impair further image processing tasks. One such example is video surveillance performed with low-cost cameras [1]. Yet, some image modalities such as computed tomography [2], [3], [4], fluorescence microscopy [5], [6], 3D computational microscopy [7], [8], electron microscopy [9] and astronomy [10], [11], in spite of their expensive sensors, can also produce noisy data due to the complexities of the data acquisition process.

The Non-Local Means (NLM) approach has been widely applied for both image [12], [13] and video denoising [14]. However, one of its major limitation is the inability to preserve local texture and small-scale structures [15]. The same occurs with other filtering techniques, such as: Bilateral Filter [16], FoF [17], KSV-D [18], SAIST [19], DDID [20], NL-Bayes [21], PID [22], Noise Clinic [23] and BM3D-SAPCA [24], the latter presenting state-of-the-art results in denoising of grayscale images [22].

To reduce the loss of details, the NLM-LBP algorithm was proposed [15]. It uses a LBP (Local Binary Pattern) descriptor [25] to modify the NLM weighting function to better account for texture regions in the 2D image domain. However, neither NLM-LBP nor BM3D-SAPCA suits higher-dimensional filtering tasks such as videos or 3D images. Although a more recent method, NLM-LBP performs poorly in such scenarios when compared with BM3D-SAPCA. Furthermore, under high noise levels scenarios, NLM-LBP tends to confuse noise with texture.

In this paper, we propose a new denoising algorithm, NLM3D-LBP-MSB, which is texture-aware and capable of handling higher-dimensional data such as videos and 3D images. We use the NLM3D as a basic denoising algorithm and follow the idea of the NLM-LBP to change the weight function with LBP algorithm. Thus, in our implementation we use a 3D version of LBP, named LBP-TOP (Local Binary Patterns on Three Orthogonal Planes) [26]. In addition, we apply an image bit plane analysis to improve the robustness of our approach to noise, which also helps preserve textures. We also propose an adaptive version of our method to better handle untextured/homogeneous regions. This version uses the distribution of the LBP non-uniform patterns to distinguish between homogeneous and textured regions and, therefore, adjust the level of denoising.

In a nutshell, the main contributions of this paper are:

- an extension of NLM-LBP [15] to handle 3D images and video data;
- the use of an image bit plane analysis to improve LBP-TOP local robustness to noise;
- an adaptive handling of textured regions.

For many of the investigated scenarios, our method has produced better results than NLM [12], NLM3D [14], NLM-LBP [15] and BM3D-SAPCA [24]. In addition, when compared to the BM3D-SPCA, the two proposed methods are simpler to understand and to implement. Moreover, they are almost as simple as the original NLM method.

II. Technical background

In this section we present the techniques used in the development of our method. First, we explain the NLM denoising method and its adaptation for video denoising, called NLM3D. We then define the LBP descriptor for images and one of its variants for video description (LPB-TOP). Finally, we describe the most significant bits (MSB) quantization, used to improve our technique robustness to noise.

A. Non-local means for image denoising

The Non-Local Means (NLM) [12] algorithm is based on the premise that images have a high degree of redundancy (similar regions) and that those regions are not always enclosed in a spatial neighborhood. Hence, NLM restores a pixel value by seeking for other pixels that have a similar neighborhood (that are in a similar patch) and not just pixels that are spatially close. In fact, in the original formulation, the search space is the entire image. Nonetheless, due to high computational cost, it is common to use the windowed NLM version, where search is limited to a smaller image region. This paper builds on the windowed NLM, hence, our discussions are limited to this version.

Given a noisy image v and a pixel i, the restored value $NL[v](i)$ of pixel i is computed as:

$$NL[v](i) = \sum_{j \in S_i} w(i,j)v(j), \tag{1}$$

where the weight $w(i,j)$ measures the similarity between pixels i and j, S_i is a $s \times s$ search window centered at pixel i and s (the window size) is a parameter of the NLM algorithm. The weight $w(i,j)$ is defined as:

$$w(i,j)\frac{1}{Z(i)}e^{-\frac{||v(\mathcal{N}_i)-v(\mathcal{N}_j)||^2_{2,a}}{h^2}}, \tag{2}$$

where $\mathcal{N}_i$ and $\mathcal{N}_j$ are $p \times p$ image regions centered at pixels i and j, respectively; p is a NLM parameter (the distance used to computed the similarity is an Euclidean distance weighted by a Gaussian kernel with standard deviation equals to a); h is another NLM parameter that defines the degree of filtering. Because the weights must sum up to one ($\sum_j w(i,j) = 1$), $Z(i)$ is a normalization term defined as follows:

$$Z(i,j) = \sum_{j \in S_i} e^{-\frac{||v(\mathcal{N}_i)-v(\mathcal{N}_j)||^2_{2,a}}{h^2}}, \tag{3}$$

A known issue with Equation 2 is that the reference pixel i tends to have an excessive weight [13]. To solve this problem, this weight is set as the maximum weight among its neighbors, according to the following equation:

$$w(i,i) = \max_{j \in S_i, j \neq i} [w(i,j)]. \tag{4}$$

The windowed NLM method for image denoising is presented in Algorithm 1.

B. Non-local means for video denoising

The NLM method was extended to process 3D data, in particular videos, by taking into account two extra features [14]. First, the pixel being denoised, denoted by i in the 2D NLM formulation, also has a third (temporal) coordinate. Second, the search window S_i has a temporal length, defined by a parameter (s_t). With that in mind, understanding and implementing the NLM for videos is fairly straightforward.

In this paper — aiming to better account for the spatio-temporal information — we used a slightly different version of the 3D NLM algorithm. It considers not only a 3D search window, but also a 3D neighborhood, while computing the similarity between two pixels. Consequently, we add an extra parameter p_t, which defines the temporal length (number of frames) of the neighborhoods $\mathcal{N}_i$ and $\mathcal{N}_j$. Besides, a

Algorithm 1 Non-Local Means Image Denoising Algorithm

```
Input: v (noisy image), s (search window size), p (patch size),
    h (denoise degree), a (kernel std)
Output: NLM[v] (restored image)
 1: for $i \in v$ do
 2:     for $j \in S_i$ do
 3:         $w(i,j) = e^{-\frac{||v(\mathcal{N}_i)-v(\mathcal{N}_j)||^2_{2,a}}{h^2}}$    ▷ Eq. 2 without $Z(i)$
 4:     end for
 5:     $w(i,i) = \max_{j \in S_i, j \neq i} [w(i,j)]$    ▷ Eq. 4
 6:     $Z(i,j) = \sum_{j \in S_i} e^{-\frac{||v(\mathcal{N}_i)-v(\mathcal{N}_j)||^2_{2,a}}{h^2}}$    ▷ Eq. 3
 7:     $w(i,i) = \frac{w(i,j)}{Z(i)}$    ▷ weight normalization
 8:     $NL[v](i) = \sum_{j \in S_i} w(i,j)v(j)$    ▷ Eq. 1
 9: end for
10: return $NLM[v]$
```

3D Gaussian kernel must be used to compute the weighted Euclidean distance.

C. Local binary patterns

Local Binary Patterns (LBP) [25] is one of the most successful approaches to describing textures. Given the good performance of the original proposal, several LBP variants and improvements followed [27], [28]. In this section, we describe the LBP version that is used in [15] and in our proposed algorithms. Such version considers uniform patterns and it is invariant to gray scale shifts, changes in scale and rotation. The explanations in this section are mainly based on [28], so the reader can check this reference for more details.

In the LBP formulation a texture T[1] for a pixel g_c is the joint distribution of g_c and a neighborhood of size P, and it is given by:

$$T = t(g_c, g_0, ..., g_{P-1}), \tag{5}$$

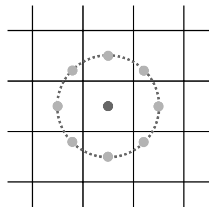
where $g_0, ..., g_{P-1}$ are the P above-mentioned neighbors. Moreover, the spatial arrangement of this neighborhood is usually circular or square. A circular neighborhood is defined by sampling P equally spaced points in a circle of radius R that is centered in g_c, as depicted in Figure 1a. Points not located at the center of a pixel need to have their values interpolated (e.g. by a bi-linear interpolation). A square neighborhood is obtained in a similar way, where the P neighbors are equally spaced over a square of side R, as depicted in Figure 1b.

Directed towards invariance to grayscale shifts, the value of g_c is subtracted for all its neighbors, as follows:

$$T = t(g_c, g_0 - g_c, ..., g_{P-1} - g_c). \tag{6}$$

This change does not cause information loss, since the original T (Equation 5) can be obtained from this new definition. Furthermore, $g_p - g_c (p = 0, 1, ..., P-1)$ are invariant to grayscale shifts. Nonetheless, g_c and, consequently, Equation 6 are not invariant to grayscale shifts. Aiming at overcoming

[1]As defined by Ojala [28]. Note that, in image processing, there is no universal definition of texture.


(a) circular neighborhood

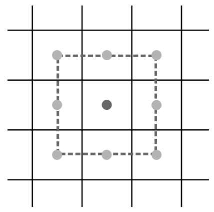

(b) square neighborhood

Fig. 1. Different types of neighborhoods for $R = 1$ and $P = 8$.

this problem, two approximations (which imply information loss) are made. First, g_c and $g_p - g_c (p = 0, 1, ..., P - 1)$ are considered independent. Second $t(g_c)$ is removed from the definitions of T. This is due to the fact that $t(g_c)$ describes the grayscale distribution within the image and does not provide much textural information. These approximations are presented in the equation bellow:

$$\begin{aligned} T &\approx t(g_c)t(g_0 - g_c, ..., g_{P-1} - g_c), \\ &\approx t(g_0 - g_c, ..., g_{P-1} - g_c). \end{aligned} \tag{7}$$

Although the texture operator presented in Equation 7 is invariant to grayscale shifts, it is not scale invariant. In order to deal with this issue, only the signs of the differences are considered:

$$T \approx t(s(g_0 - g_c), ..., s(g_{P-1} - g_c)), \tag{8}$$

where the sign function s is given by:

$$s(n) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0. \end{cases} \tag{9}$$

As the distribution T is represented by zeros and ones, the first version of the LBP operator (code) can be defined as follows:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p. \tag{10}$$

As such, $LBP_{P,R}$ achieves invariance regarding both scale and shifts in grayscale.

To reach invariance to rotation, a new operator ($LBP^{ri}_{P,R}$) is defined as the smallest code (natural number) that can be obtained by performing circular bit-wise operations on $LBP_{P,R}$. The new operator is then defined as:

$$LBP^{ri}_{P,R} = \min\{ROR(LBP_{P,R}, i) \mid i = 0, ..., P - 1\}, \tag{11}$$

where $ROR(l, m)$ is a function that performs l circular bit-wise right shifts on m.

Up to this stage, the operator $LBP^{ri}_{P,R}$ is invariant to grayscale shifts, changes in scale and rotation. However, it was observed that most of the $LBP^{ri}_{P,R}$ transitions rarely occurs, while just a few accounts for more than 90% of the observations in some cases [29]. Yet, the most common patterns were the ones which, when considered circularly, exhibited two bit transitions or less (from 0 to 1 or from 1 to 0). These patterns were named uniforms. This concept is illustrated in Figure 2, where the 8 possible uniform patterns for a circular neighborhood ($P = 8$ and $R = 1$) are presented.

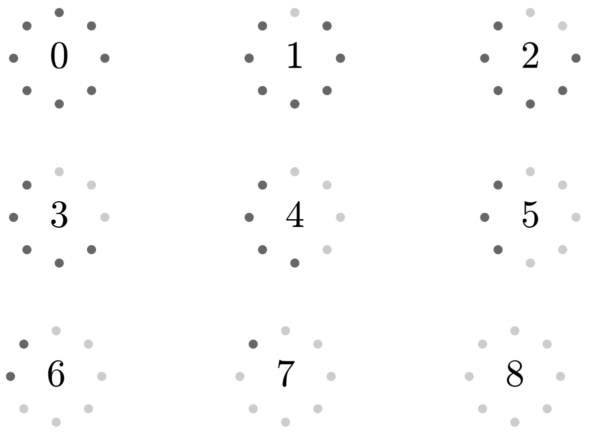


Fig. 2. All possible uniform patterns for a circular neighborhood of $R = 1$ and $P = 8$. The patterns 0 and 8 do not have any transitions, while the rest of the patterns have two transitions. This image is based on Figure 2 of [28].

Finally, with the LBP code of each pixel, we can compute the LBP descriptor: a histogram in which each uniform pattern has its own bin and the non-uniform patterns are all mapped onto a single extra bin. The histogram can either be computed for the whole image or for image cells. In the latter, histograms of all cells are concatenated to generate the final feature vector.

D. LBP-TOP descriptor

The LBP-TOP (Local Binary Patterns on Three Orthogonal Planes) descriptor [26] is a simple LBP extension for describing videos. By considering a video as a function $f(x, y, t)$ – with x and y representing a 2D spacial position and t a temporal coordinate – a 2D LBP descriptor is computed for every plane XY, XT and YT within the video. Next, three histograms are computed, one for each set of planes (XY, XT and YT). The final descriptor is obtained by concatenating these three histograms. This process is shown in Figure 3. As in the 2D LBP version, histograms are not computed over the whole video. We first divide it into cuboids, run LBP-TOP on each of them and finally concatenate all histograms to produce the final feature vector.

E. MSB image quantization

The Most Significant Bits (MSB) quantization has several applications in image and video processing. For instance, [30] uses MSB to quantize color images and, as a consequence, reduce the dimensionality of extracted features. In our case, we use MSB image analysis to discard the less significant bits of an image, as they have a higher chance of being affected by noise.

The MSB quantization of a 8-bit pixel can be defined as follows:

$$\hat{p} = p \wedge \neg(2^{8-n} - 1), \tag{12}$$

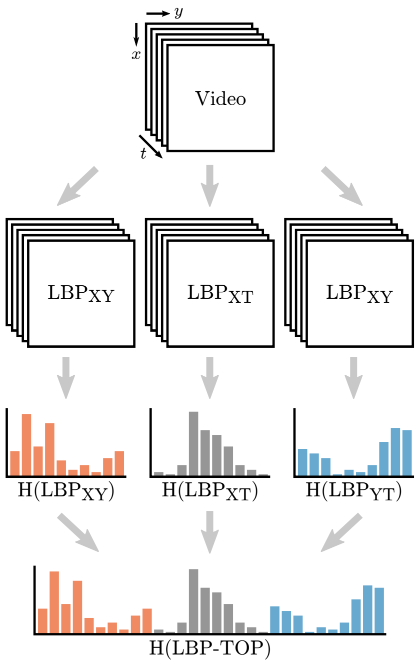


Fig. 3. Main steps of the LBP-TOP descriptor.

where $\hat{p}$ is the output (quantized) image, p is the input image and n is the number of most significant bits from p that will be in $\hat{p}$.

As an example, consider $p = 221$ (binary `11011101`) and $n = 5$. Computing $2^{8-n} - 1$ results in 7 (binary `00000111`). Then, by inverting ($\neg$) 7, 248 (binary `11111000`) is obtained. Finally, the bitwise `and` operation ($\wedge$) yields $\hat{p} = 216$ (binary **`11011`**`000`). In this example, the 5 most significant bits of p are kept in $\hat{p}$, while the other 3 are set to zero.

F. NLM-LBP image denoising method

The NLM variant presented in [15] (NLM-LBP) aims at better preserving textures and details in image denoising. Such algorithm, used in our proposed approach, introduces the following weighting function:

$$w_m(i,j) = w_{NLM}(i,j) \times w_{LBP}(i,j), \qquad (13)$$

where $w_{NLM}(i,j)$ is the original NLM weighting function (Equation 2) and $w_{LBP}(i,j)$ is based on the LBP description (histogram) of the neighborhoods $\mathcal{N}_i$ and $\mathcal{N}_j$. The weight $w(i,j)_{LBP}$ is then defined as:

$$w_{LBP}(i,j) = \frac{1}{Z(i)} e^{-\frac{D(H(L_i),H(L_j))}{h(S_i)}}, \qquad (14)$$

where $h(S_i)$ is the standard deviation of the pixel values in the search window S_i; L_i and L_j are, respectively, the LBP codes for the neighborhoods $\mathcal{N}_i$ and $\mathcal{N}_j$; $H(L_i)$ and $H(L_j)$ are the LBP descriptions (histograms) for $\mathcal{N}_i$ and $\mathcal{N}_j$; $Z(i)$ is a normalization factor and D is the Chi-square distance computed as:

$$D(H(L_i), H(L_j)) = \sum_{n=1}^{B} \frac{(H(L_i)_n - H(L_j)_n)^2}{H(L_i)_n + H(L_j)_n}, \qquad (15)$$

where n represents a histogram bin index and B is the histogram size.

Lastly, two remarks are necessary: a) $w_m(i,i)$ needs to be adjusted to avoid excessive weight, as in the original NLM definition (Equation 4) and b) $w_m(i,j)$ has to be normalized in order to guarantee sum equal 1.

III. Proposed Methods

This section introduces our two NLM variants: NLM3D-LBP-MSB and NLM3D-LBP-Adaptive. NLM3D-LBP-MSB extends NLM-LPB and is suitable for processing videos. Moreover, unlike NLM-LBP, this method employs an MSB quantization step to minimize the influence of noise in its weighting function. The second method, NLM3D-LBP-Adaptive, labels each pixel as being part of either a uniform or a textured region. To that, we use LBP-TOP. Pixel restoration is finally achieved by taking into account the type of region they are in.

A. NLM3D-LBP-MSB

NLM3D-LBP-MSB is a two-stage approach: pre-processing and pixel denoising. Pre-processing begins with an MSB quantization of each video frame. This quantization preserves the m most significant bits, where m is a parameter of our algorithm. The MSB video is then used to compute the LBP-TOP patterns (codes) for three orthogonal planes (XY, XT and YT), yielding three new videos: LBP$_{XY}$, LBP$_{XT}$ and LBP$_{YT}$. As our experiments will show, performing denoising over MSB videos, as opposed to the original data, produces better results.

The denoising stage takes the three LBP-TOP videos generated during pre-processing to compute the following weighting function:

$$w_M(i,j) = w_{NLM3D}(i,j) \times w_{TOP}(i,j), \qquad (16)$$

where $w_{NLM3D}(i,j)$ is the weighting function from NLM3D (see Section II-B) and $w_{TOP}(i,j)$ is our novel 3D extension defined as:

$$w_{TOP}(i,j) = \frac{1}{Z(i)} e^{-\frac{D(H(L_i^{TOP}),H(L_j^{TOP}))}{h(S_i)}}, \qquad (17)$$

where $h(S_i)$ is the standard deviation of the pixels values in the search window S_i (in the MSB video); L_i^{TOP} and L_j^{TOP} are, respectively, the LBP-TOP codes for the 3D neighborhoods of pixels i and j; $H(L_i^{TOP})$ and $H(L_j^{TOP})$ are the LBP-TOP descriptions (histograms) of L_i^{TOP} and L_j^{TOP}; $Z(i)$ is a normalization factor and D is the Chi-square distance.

From this point onwards our algorithm is the same as NLM3D: we use w_M as weights for the original pixel values (not the MSB ones). The diagram of Figure 4 illustrates our method.

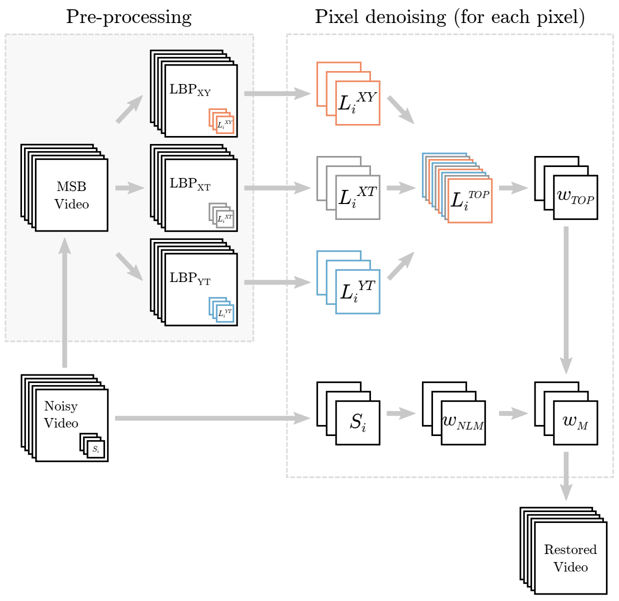


Fig. 4. NLM3D-LBP-MSB diagram.

B. NLM3D-LBP-Adaptive

The NLM3D-LBP-MSB method employs an LBP-TOP descriptor to compute the NLM3D weight matrix. However, for untextured/homogeneous regions with high levels of Gaussian white noise, the LBP descriptor tends to interpret noise as texture. As a consequence, the algorithm will preserve such noisy regions. To overcome this limitation, we proposed an adaptive version, named NLM3D-LBP-Adaptive.

This adaptive version looks at the sum of LBP histogram last bin of each plane, which contains the non-uniform patterns, as a means to compute the difference between texture and noise. To that, we introduce a threshold τ. If the sum of the non-uniform bins is higher than τ, pixels are filtered with the NLM3D weight matrix and noise is filtered out. Otherwise, the weight matrix computed with LBP-TOP and NML3D methods will be employed. In this case, the region is interpreted as texture and its content preserved. In other words, this adaptive version is capable of filtering out homogeneous regions with high Gaussian noise levels, while preserving texture.

IV. Experiments and results

Considering that our main goal is to improve the restoration results of non local video denoise, we compare both proposed methods, NLM3D-LBP-MSB and NLM3D-LBP-Adaptive, with the following NLM variants: the original 2D and 3D NLM, and NLM-LBP. We also included the BM3D-SAPCA method in the comparison, since it is a state-of-the-art image denoising method.

We begin by defining our test sequences and our test methodology. Then we perform a grid search to find the best parameters for each method in each sequence. We also analyse the performance impacts of the MSB quantization. Lastly, we compare the above-mentioned denoising methods by means of the PSNR and SSIM measures and visual analysis of the restored images.

A. Test frames

Six 320×180 noise-free RGB video sequences were taken from "The Big Buck Bunny video"[2] and then converted to grayscale. The central frames of each video sequence contains a lot of small details, as can be seen in Figure 5.

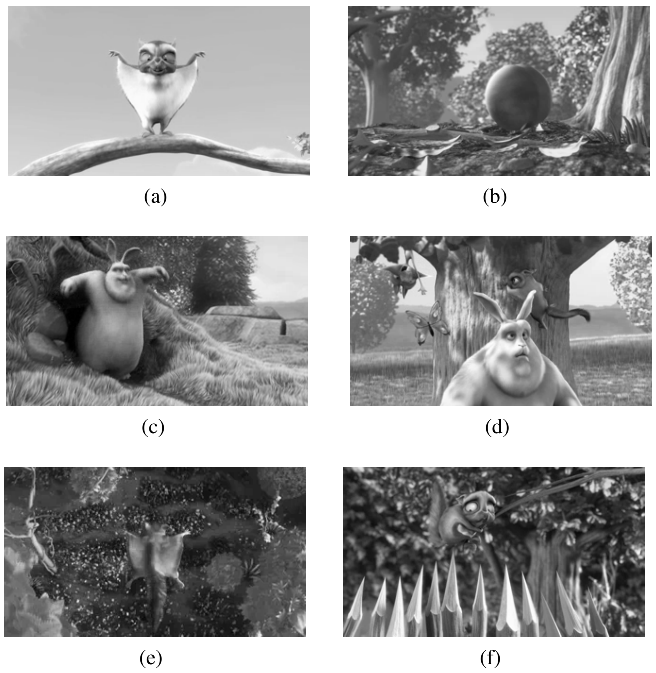


Fig. 5. Noise-free grayscale central frames for the six videos: frames (a) to (f) are referred to as sequences 1 to 6, respectively.

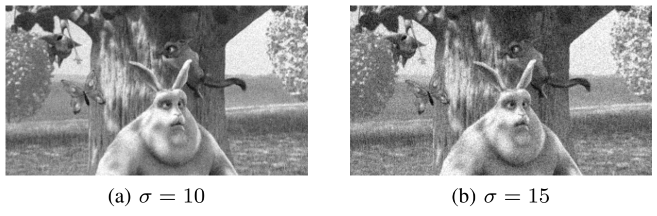
We then corrupted all grayscale videos with 4 different levels of Gaussian white noise ($\sigma = \{10, 15, 20, 25\}$), yielding 24 noisy video sequences as shown in Figure 6. Finally, we evaluated the above-mentioned methods by performing denoising on the central frame only for all the 24 video sequences. Methods are compared using the full reference quality measures PSNR and SSIM [31].

B. Parameter analysis

A grid search was performed in order to find the best parameter combination for each method in each test sequence. Such procedure allows for an unbiased comparison, as we compare the best result of each method. The parameters were set as follows: search window size $s = \{7, 9, 11, 13\}$, patch size $p = 5$ and denoise degree $h = \{10, 15, 20, 25\}$. We also considered the pair $s = 5$ and $p = 3$ combined with all h values. As for the 3D methods, the parameters s_t (the temporal length of the search window) and p_t (the temporal length of the neighborhoods) were set as s and p, respectively.

Other parameters were defined as follows: the number of the most significant bits $m = 3$ (for the proposed NLM3D-LBP-MSB method), the threshold $\tau = 0.09$ (for the proposed

[2]This video is licensed under the Creative Commons Attribution 3.0 license. The original (colored) 320×180 video can be download at: http://download.blender.org/peach/bigbuckbunny_movies/BigBuckBunny_320x180.mp4



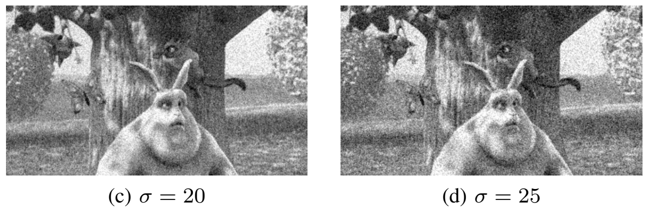


Fig. 6. Different noise levels for the central frame of Sequence 4 (Fig. 5d).

NLM3D-LBP-Adaptive method), the NLM Gaussian kernel std $a = 1$. The LBP and LBP-TOP descriptors used a square neighborhood of size $P = 8$ and radius $R = 1$.

Amongst all parameters, the search window size (s) was the one with the highest impact on the denoising outcome. Figures 7 and 8 exemplify such influence for video Sequence 4 and two different levels of noise. Changes in remaining parameters did not alter denoising significantly.

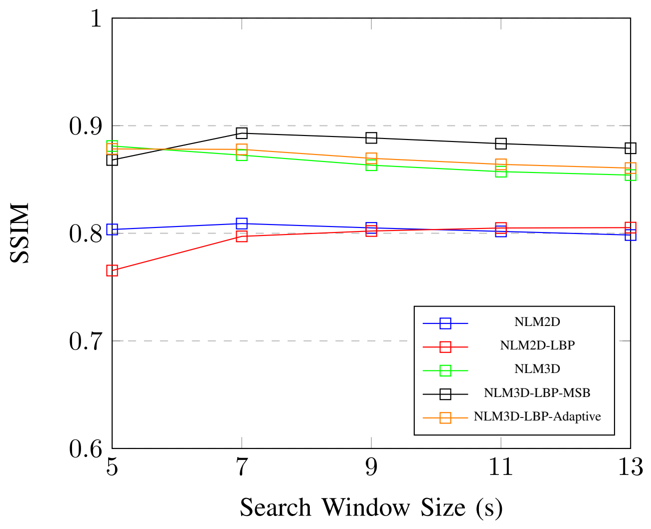


Fig. 7. SSIM results for different values of s for Sequence 4 with $\sigma = 20$.

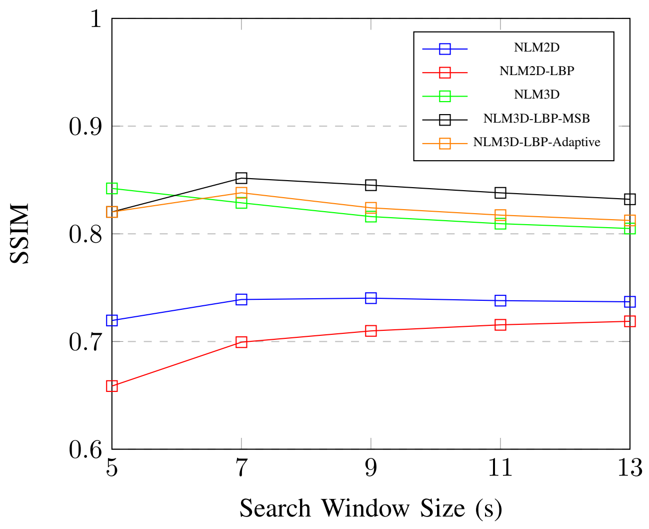


Fig. 8. SSIM results for different values of s for Sequence 4 with $\sigma = 25$.

C. MSB performance impact

We analysed the impact of the MSB on the performance of our method by running and comparing the restoration with and without it. Results are shown in Table I. The use of MSB increased the denoising performance in 22 out of 24 cases, for both the PSNR and SSIM measures.

It is worth mentioning that the MSB quantization step is computationally cheap, especially if compared to the cost of the NLM method itself, and does not add to the overall complexity of the denoising process. Hence, despite the marginal increase on PSNR and SSIM measures, the addition of the MSB strategy to our method proved very advantageous.

TABLE I
MSB PERFORMANCE GAIN FOR THE PROPOSED METHOD.

Noise (σ)		With MSB		Without MSB		Gain	
		PSNR	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**
Sequence 1	**10**	39.3453	0.9773	39.1824	0.9757	0.1629	0.0016
	15	35.6766	0.9569	35.4909	0.9551	0.1857	0.0018
	20	34.6349	0.9176	34.4995	0.9152	0.1354	0.0024
	25	33.1419	0.8195	32.9982	0.8237	0.1437	-0.0042
Sequence 2	**10**	35.8919	0.9667	35.8154	0.9659	0.0765	0.0008
	15	32.8849	0.9344	32.8305	0.9328	0.0544	0.0016
	20	30.8613	0.897	30.8292	0.8959	0.0321	0.0011
	25	29.3623	0.8623	29.3175	0.8624	0.0448	-0.0001
Sequence 3	**10**	34.5067	0.9456	34.4889	0.9452	0.0178	0.0004
	15	32.1897	0.9065	32.1495	0.9042	0.0402	0.0023
	20	30.213	0.8604	30.1866	0.8584	0.0264	0.002
	25	28.8755	0.817	28.8803	0.8159	-0.0048	0.0011
Sequence 4	**10**	35.9273	0.9592	35.8471	0.9581	0.0802	0.0011
	15	33.2705	0.9266	33.2385	0.9247	0.032	0.0019
	20	31.2472	0.893	31.2223	0.8913	0.0249	0.0017
	25	29.6398	0.8517	29.6388	0.8509	0.001	0.0008
Sequence 5	**10**	32.3515	0.9116	32.3551	0.9114	-0.0036	0.0002
	15	30.056	0.8503	30.0098	0.8492	0.0462	0.0011
	20	28.2856	0.7916	28.2468	0.7908	0.0388	0.0008
	25	27.0759	0.7377	27.0529	0.7369	0.023	0.0008
Sequence 6	**10**	36.7086	0.9789	36.5899	0.9782	0.1187	0.0007
	15	33.8042	0.9598	33.7611	0.9583	0.0431	0.0015
	20	31.7354	0.9369	31.6173	0.9342	0.1181	0.0027
	25	30.2572	0.9155	30.173	0.9118	0.0842	0.0037

D. Quantitative evaluation

We computed the PSNR and SSIM and compared our techniques with the original 2D and 3D NLM, NLM-LBP and BM3D-SAPCA for each of the 24 noisy video sequences. Table II presents the best results produced with the grid search as explained in the beginning of this section.

The proposed NLM3D-LBP-MSB method produced the best results (both SSIM and PSNR) in the majority of the cases, from video sequences 2 to 6. As for sequence 1, the proposed adaptive method was best in all cases, according to

TABLE II

RESULTS AND COMPARISONS BETWEEN THE METHODS OF DENOISING.

Method		$\sigma = 10$		$\sigma = 15$		$\sigma = 20$		$\sigma = 25$	
		PSNR	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**
Sequence 1	NLM2D	36.4860	0.9607	34.2694	0.9190	32.3205	0.7978	29.5508	0.6157
	NLM2D-LBP	36.3286	0.9567	34.5121	0.8958	31.6209	0.7463	28.4030	0.5533
	NLM3D	39.1067	0.9761	35.4872	0.9548	34.2346	0.9204	33.2680	0.8373
	NLM3D-LBP-Adaptive	**39.6609**	0.9780	**36.1827**	0.9580	**34.7596**	0.9247	**33.3100**	0.8498
	NLM3D-LBP-MSB	39.3453	0.9773	35.6766	0.9569	34.6349	0.9176	33.1419	0.8195
	BM3D-SAPCA	37.7761	**0.9804**	33.9791	**0.9711**	33.1466	**0.9636**	32.2095	**0.9517**
Sequence 2	NLM2D	31.5446	0.9250	29.4132	0.8725	27.6455	0.8116	26.2403	0.7599
	NLM2D-LBP	31.9392	0.9282	29.5394	0.8747	27.7995	0.8125	25.8709	0.7450
	NLM3D	35.5902	0.9612	32.2032	0.9257	30.3524	0.8887	28.8799	0.8541
	NLM3D-LBP-Adaptive	35.6238	0.9633	32.5278	0.9273	30.4806	0.8885	29.0036	0.8533
	NLM3D-LBP-MSB	**35.8919**	**0.9667**	**32.8849**	**0.9344**	**30.8613**	**0.8970**	**29.3623**	**0.8623**
	BM3D-SAPCA	28.1310	0.9294	25.1850	0.8862	24.6264	0.8520	24.2287	0.8162
Sequence 3	NLM2D	31.8570	0.9025	29.4976	0.8375	28.0066	0.7769	26.5312	0.7076
	NLM2D-LBP	31.8299	0.9021	29.5623	0.8386	27.8893	0.7739	26.0708	0.6885
	NLM3D	33.9603	0.9392	31.6472	0.8964	29.8590	0.8509	28.5912	0.8084
	NLM3D-LBP-Adaptive	34.2582	0.9410	31.7607	0.8968	29.8679	0.8499	28.5863	0.8058
	NLM3D-LBP-MSB	**34.5067**	**0.9456**	**32.1897**	**0.9065**	**30.2130**	**0.8604**	**28.8755**	**0.8170**
	BM3D-SAPCA	30.4885	0.9167	29.2995	0.8667	28.4257	0.8250	27.6300	0.7851
Sequence 4	NLM2D	32.1658	0.9120	29.9285	0.8565	28.3054	0.8090	26.7817	0.7403
	NLM2D-LBP	32.3046	0.9152	30.0325	0.8598	28.2387	0.8053	26.1908	0.7188
	NLM3D	35.1174	0.9510	32.5785	0.9148	30.6604	0.8811	29.2038	0.8421
	NLM3D-LBP-Adaptive	35.5837	0.9530	32.8514	0.9148	30.4716	0.8791	29.1446	0.8381
	NLM3D-LBP-MSB	**35.9273**	**0.9592**	**33.2705**	**0.9266**	**31.2472**	**0.8930**	**29.6398**	**0.8517**
	BM3D-SAPCA	32.1545	0.9294	30.6803	0.8941	29.4789	0.8614	28.5182	0.8294
Sequence 5	NLM2D	30.4781	0.8838	28.4314	0.8152	27.1071	0.7537	25.8679	0.6824
	NLM2D-LBP	30.9665	0.8884	28.9446	0.8226	27.3502	0.7571	25.7680	0.6755
	NLM3D	31.3226	0.8961	29.9573	**0.8521**	28.5034	0.7884	26.8207	0.7266
	NLM3D-LBP-Adaptive	31.6860	0.9013	29.8284	0.8502	28.4462	0.7884	26.8705	0.7239
	NLM3D-LBP-MSB	**32.3515**	**0.9116**	30.0560	0.8503	28.2856	**0.7916**	27.0759	**0.7377**
	BM3D-SAPCA	32.3485	0.8991	**30.2467**	0.8412	**28.7423**	0.7877	**27.5169**	0.7334
Sequence 6	NLM2D	32.3499	0.9502	29.9921	0.9166	28.1659	0.8790	26.4796	0.8231
	NLM2D-LBP	32.5187	0.9501	29.9574	0.9129	27.8302	0.8673	25.9376	0.8055
	NLM3D	35.9805	0.9754	33.2654	0.9556	31.3227	0.9327	29.8373	0.9106
	NLM3D-LBP-Adaptive	36.4736	0.9764	33.5822	0.9541	31.1946	0.9320	29.6042	0.9071
	NLM3D-LBP-MSB	**36.7086**	**0.9789**	**33.8042**	**0.9598**	**31.7354**	**0.9369**	**30.2572**	**0.9155**
	BM3D-SAPCA	33.5364	0.9559	31.5411	0.9293	30.0891	0.9011	29.0055	0.8773

PSNR measure, while the BM3D-SAPCA produced the best results with the SSIM measure.

One should notice that sequence 1 (Figure 5a) possesses broader homogeneous regions and far less texture and small details than sequences 2 to 6. This is an indication of the ability of the NLM3D-LBP-MSB method in handling texture and small details. When data is predominantly composed of homogeneous regions, the proposed adaptive method, as well as the BM3D-SAPCA algorithm, are more adequate.

E. Qualitative evaluation by visual analysis

Figure 9 illustrates a zoomed-in region of a frame processed with all denoising methods. Textured regions (tree trunk) and detailed structures (butterfly wings) are particularly better preserved in our proposed NLM3D-LBP-MSB algorithm. However, for flat regions (background sky and grass) the BM3D-SAPCA (Figure 9f), NLM3D (Figure 9e) and our adaptive version (Figure 9g) are better. Nonetheless, the quantitative analysis provided in Table II also reveals that the adaptive version is only outperformed by its non-adaptive counterpart, for the majority of the textured video sequences (2, 3, 4 and 6). Hence, the adaptive version provides a good trade-off for filtering videos that combine texture and homogeneous regions.

V. REPRODUCIBILITY REMARKS

For reproducibility purposes, both code and video sequences used in our experiments are available on-line[3].

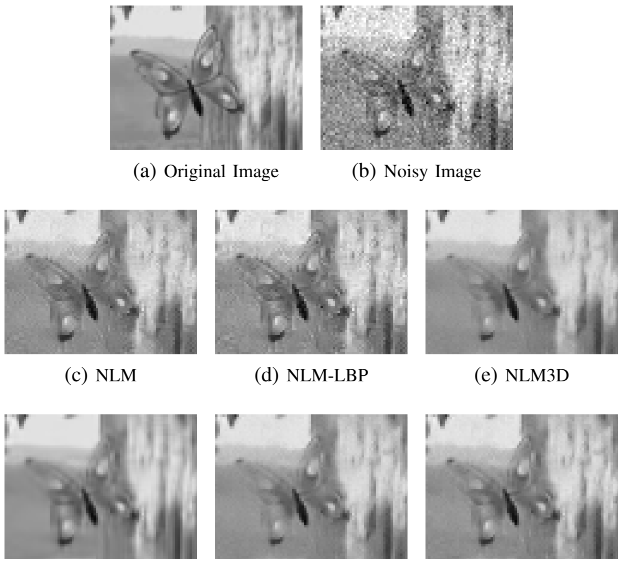
VI. CONCLUSIONS

We introduced two new denoising methods that aim at restoring videos and 3D images while maintaining texture information and small structures. Both methods use the LBP-TOP descriptor to account for texture. The first approach (NLM3D-LBP-MSB) employs a MSB quantization to improve robustness to noise, while the second (NLM3D-LBP-Adaptive) analyses the distribution of non-uniform LBP patterns to distinguish textured from homogeneous regions.

We compared the proposed algorithms with several NLM variants and the BM3D-SAPCA method. To that, we used 6 different videos sequences corrupted with 4 different levels of additive white Gaussian noise. Qualitative (visual inspection) as well as quantitative (SSIM and PSNR measures) experiments have been provided.

Our results indicate that both proposed methods are capable of restoring videos while preserving textures and small details. We have also shown that MSB quantization can boost robustness to noise. Finally, our experiments suggest that LBP

[3]Repository url: https://github.com/welintonandrey/3d_denoising


(a) Original Image (b) Noisy Image

(c) NLM (d) NLM-LBP (e) NLM3D

(f) BM3D-SAPCA (g) NLM3D-LBP-Adaptive (h) NLM3D-LBP-MSB

Fig. 9. Visual comparison of the denoising methods.

non-uniform patterns are an effective approach to detecting textured and homogeneous regions in denoising tasks.

For future work, we intend to better analyse the value parameter τ on the detection of homogeneous regions, to combine the BM3D-SAPCA method with the LBP descriptor to change the block matching process and to apply the proposed methods on unconstrained videos.

Acknowledgment

This work was supported by FAPESP (grants #2014/21888-2, #2015/04883-0 and #2015/05310-3).

References

[1] M. J. Roshtkhari and M. D. Levine, "An on-line, real-time learning method for detecting anomalies in videos using spatio-temporal compositions." *Computer Vision and Image Understanding*, vol. 117, no. 10, pp. 1436–1452, 2013.

[2] F. Attivissimo, G. Cavone, A. M. L. Lanzolla, and M. Spadavecchia, "A technique to improve the image quality in computer tomography," *IEEE Transactions on Instrumentation and Measurement*, vol. 5, no. 59, pp. 1251–1257, 2010.

[3] Y. Lou, X. Zhang, S. Osher, and A. Bertozzi, "Image recovery via nonlocal operators," *Journal of Scientific Computing*, vol. 42, no. 2, pp. 185–197, 2010.

[4] J. Wang, T. Li, H. Lu, and Z. Liang, "Penalized weighted least-squares approach to sinogram noise reduction and image reconstruction for low-dose x-ray computed tomography," *IEEE Transactions on Medical Imaging*, vol. 25, no. 10, pp. 1272–1283, 2006.

[5] J. Boulanger, C. Kervrann, P. Bouthemy, P. Elbau, J.-B. Sibarita, and J. Salamero, "Patch-based nonlocal functional for denoising fluorescence microscopy image sequences," *IEEE Transactions on Medical Imaging*, vol. 29, no. 2, pp. 442–454, 2010.

[6] S. Delpretti, F. Luisier, S. Ramani, T. Blu, and M. Unser, "Multiframe SURE-let denoising of timelapse fluorescence microscopy images," in *Proceedings of the Fifth IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI'08)*, 2008, pp. 149–152.

[7] M. Ponti, E. S. Helou, P. J. S. G. Ferreira, and N. D. A. Mascarenhas, "Image restoration using gradient iteration and constraints for band extrapolation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 1, pp. 71–80, Feb 2016.

[8] M. Ponti-Jr, N. D. Mascarenhas, P. J. Ferreira, and C. A. Suazo, "Three-dimensional noisy image restoration using filtered extrapolation and deconvolution," *Signal, Image and Video Processing*, vol. 7, no. 1, pp. 1–10, 2013.

[9] W. Jiang, M. L. Baker, Q. Wu, C. Bajaj, and W. Chiu, "Applications of a bilateral denoising filter in biological electron microscopy," *Journal of structural biology*, vol. 144, no. 1, pp. 114–122, 2003.

[10] S. Beckouche, J.-L. Starck, and J. Fadili, "Astronomical image denoising using dictionary learning," *Astronomy & Astrophysics*, vol. 556, p. A132, 2013.

[11] M. Mäkitalo and A. Foi, "Optimal inversion of the anscombe transformation in low-count poisson image denoising," *IEEE Transactions on Image Processing*, vol. 20, no. 1, pp. 99–109, 2011.

[12] A. Buades, B. Coll, and J. Morel, "A non-local algorithm for image denoising," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005, pp. 60–65.

[13] ——, "Non-Local Means Denoising," *Image Processing On Line*, vol. 1, 2011.

[14] ——, "Denoising image sequences does not require motion estimation," in *Advanced Video and Signal Based Surveillance, 2005. AVSS 2005. IEEE Conference on*. IEEE, 2005, pp. 70–74.

[15] F. M. Khellah, "Application of local binary pattern to windowed nonlocal means image denoising," in *17th International Conference on Image Analysis and Processing - ICIAP 2013*, Naples, Italy, 2013, pp. 21–30.

[16] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Computer Vision, 1998. Sixth International Conference on*. IEEE, 1998, pp. 839–846.

[17] S. Roth and M. J. Black, "Field of Experts: A Framework for Learning Image Priors," *IEEE Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 860—-867, 2005.

[18] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[19] W. Dong, G. Shi, and X. Li, "Nonlocal image restoration with bilateral variance estimation: A low-rank approach," *IEEE Transactions on Image Processing*, vol. 22, no. 2, pp. 700–711, 2013.

[20] C. Knaus and M. Zwicker, "Dual-domain image denoising," *2013 IEEE International Conference on Image Processing, ICIP 2013 - Proceedings*, no. 4, pp. 440–444, 2013.

[21] M. Lebrun, A. Buades, and J. Morel, "A nonlocal Bayesian image denoising algorithm," *SIAM Journal on Imaging Sciences*, vol. 6, no. 3, pp. 1665–1688, 2013.

[22] C. Knaus and M. Zwicker, "Progressive image denoising," *IEEE Transactions on Image Processing*, vol. 23, no. 7, pp. 3114–3125, 2014.

[23] M. Lebrun, M. Colom, and J.-m. Morel, "The Noise Clinic : a Blind Image Denoising Algorithm A Generalized Nonlocal Bayesian Algorithm," *Image Processing On Line*, vol. 5, pp. 1–54, 2015.

[24] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Bm3d image denoising with shape-adaptive principal component analysis." in *SPARS'09-Signal Processing with Adaptive Sparse Structured Representations*, Saint Malo, France, 2009.

[25] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *ICPR94*, 1994, pp. A:582–585.

[26] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 915–928, Jun. 2007.

[27] L. Nanni, A. Lumini, and S. Brahnam, "Survey on LBP based texture descriptors for image classification," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3634–3641, Feb. 2012.

[28] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[29] M. Pietikäinen, T. Ojala, and Z. Xu, "Rotation-invariant texture classification using feature distributions," *Pattern Recognition*, vol. 33, pp. 43–52, 2000.

[30] M. Ponti, T. S. Nazaré, and G. S. Thumé, "Image quantization as a dimensionality reduction procedure in color and texture feature extraction." *Neurocomputing*, vol. 173, pp. 385–396, 2016.

[31] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.