# SUPPRESSION OF MOSQUITO NOISE BY RECURSIVE EPSILON-FILTERS

*Houssam Abbas and Lina J. Karam*

Department of Electrical Engineering
Arizona State Univeristy
Tempe, Arizona 85287-5706
{houssam.abbas, karam}@asu.edu

## ABSTRACT

This work addresses the problem of Mosquito Noise (MN) reduction in compressed video sequences. A compression-blind approach is adopted; the advantage of such an approach is that it is independent of the particular compressor used and of its particular settings. A recursive filtering scheme is presented. It is shown how the filtering parameter $\epsilon$ can be adaptively selected to maximize the denoising performance by minimizing the number of outlier pixels in the filter's support. Simulation results show that the proposed blind MN-denoising scheme outperforms existing MN-denoising methods.

***Index Terms—*** Mosquito Noise, Epsilon Filter, Video, Denoising, Restoration

## 1. INTRODUCTION

Standard video compression algorithms (e.g., MPEG-2, MPEG-4, H.264) induce a number of common distortions or "artifacts" on the image: these include blocking, ringing, and mosquito noise. Blocking is characterized by adjacent blocks having (visually) significantly different average intensity levels. Ringing denotes the appearance of duplicate, lesser intensity edges parallel to true edges. The further away the fake edges from the true one, the less pronounced they become. Finally, mosquito noise (MN), which is the focus of this paper, denotes the appearance of flickering "clouds" of pixels around borders of moving objects. Thus it also has a temporal nature as well as a spatial one. Each of these artifacts presents a visual annoyance to the human viewer, especially in High Definition content. Numerous algorithms have been proposed to reduce these artifacts, from general de-noising procedures designed to address all three [1], to artifact-specific de-noising algorithms [2]. One may also classify de-noising algorithms as either compression-aware, to indicate that the algorithm depends on knowledge/estimation of the compressor's parameters, or compression-blind, to indicate that the algorithm does not use this information.

The focus in the current paper is on MN and the development of a fast, flexible, compression-blind algorithm to reduce it. The advantage of a compression-blind approach is that it is independent of the particular compressor used and its particular settings.

Compared to blocking and ringing, MN has received the least attention in the literature, be it on the analysis or on the de-noising side. This is despite the fact that MN can cause significant visual annoyance due to its temporal nature. One of the main difficulties in dealing with MN is that it is still ill-defined, and there is no consensus on how it is generated and how it manifests itself. Another difficulty stems from the fact that MN is not generated by a single clear mechanism but rather by a combination of different mechanisms. For example, while quantization of basis coefficients is known to be one of the causes of MN [3, 4, 5], MN is further characterized in [3] as a form of moving ringing that appears in 'dynamic scenes' at medium bit-rates. In [5], MN is characterized as 'a time-dependent video compression impairement in which the high-frequency spatial detail in video images having crisp edges is aliased intermittently'. In addition, the authors of [5] indicate that they observed MN in still image compression. In [4], MN is attributed to 'the coarseness of the quantization of DCT coefficients of motion-compensated block residuals (in the case of coding in INTER mode).' This does not explain the appearence of MN in *still* image compression reported in [5]. Finally, some researchers suggest that the varying bit-rate allocation from frame to frame plays a role in the creation of MN. From the above, it can be seen that MN is the result of many interacting compression mechanisms, including quantization, motion compensated prediction and varying bit-rate allocation. These same mechanisms are responsible for other observed types of artifacts, such as blocking and ringing, which makes it hard to analyse and de-noise MN in isolation of other artifacts.

This work was funded in part by Intel Corp.

This paper is organized as follows. Section 2 describes the existing MN de-noising techniques. A new ompression-blind MN denoising scheme is presented in Section 3. Simulation results and comparisons with existing techniques are presented in Section 4.

## 2. EXISITING MOSQUITO DE-NOISING METHODS

As indicated in Section 1, there are two main classes of MN-denoising methods: compression-aware, which we won't review here, and compression-blind ( [6, 7, 8, 9, 10, 11, 12, 4, 13]). In this work, the focus is on conpression-blind MN-denoising. Most of the MN-specific literature emphasizes the fact that MN is a small-amplitude deviation around a mean, usually set to 0. In [6], a spatial median filter is applied to the DC coefficients of the DCT block-transforms, followed by a temporal median filter on those coefficients. Blocks are designated as motion or motionless based on a motion threshold T, and only motionless blocks are filtered; the rationale for this is that moving areas are visually more significant than others, and hence, one should avoid blurring them by the temporal median. This is rather a shortcoming of this method (called MNR by its developers in [6]) since, as observed, MN also appears near edges of *moving* objects, so excluding them degrades the performance. This highlights one of the main challenges in suppressing MN: its spatial proximity to the edges and moving features means that one should be careful about not distorting these features.

A compression-blind approach is proposed in [7]: intensity blocks are classified as edge-blocks, if they contain an edge, and as non-edge blocks otherwise. It is assumed that only edge-blocks exhibit MN, and hence only these are filtered. Pixels that are not edge pixels are then filtered using a classical local Wiener filter, whose support *could* contain edge pixels. However, in [7], MN is identified with ringing; it is well known however that MN and ringing are separate artifacts, though the latter contributes to the former. Another issue is that non-edge blocks *can* contain MN, but they are not filtered in [7]. In addition, the derivation of the Wiener filter assumes zero correlation between the clean signal and the noise; given the dependence of MN on content, this is not valid. Finally, the derivation of the Wiener filter also assumes stationarity of all signals involved; the fact that no care is taken to exclude edge pixels from the filter's support means that the stationarity assumption is violated. Henceforth, we refer to this method as LOWTEM, for LOcal Wiener filter, TEMporal filtering.

Another compression-blind paradigm is presented in [8], using spatial median filtering. In this method, edge pixels are detected and *not* processed, nor are they included in the filtering support of other pixels. However, the spatial median used in [8] results in images with blurred texture and is intended to be applied for de-noising very low bit-rate video (8-40 kbps), with low-quality texture [8]. This is not true for the typical medium- to high-bitrate content. So the post-processor in [8] is not applicable for our purposes, since our interest is in a de-noising technique that works at all rates.

One main challenge of MN de-noising is that texture itself exhibits a small fluctuation around a mean, similar to MN. So, *it is hard to decide categorically whether the fluctuation within the current processing aperture is clean texture or MN*. To address this issue, Arakawa et al. introduced $\epsilon$-*filters* [9], and various modifications ( [9, 10, 11, 12, 4, 13]) were proposed to enhance their performance. An $\epsilon$-filter is a point-wise low-pass filter with space-varying support. Only pixels whose intensity is within an $\epsilon$ of the pixel currently being filtered are included in the support. A simple extension to the time-domain is made in [10]. The filter's performance depends significantly on the value of $epsilon$. The works in [4, 14, 13] suggest (with slight variation between them) that $\epsilon$ be computed off-line on a database of images so as to minimize the average Mean Squared Error (MSE) between filtered and clean images. However, an adaptive $\epsilon$ that minimizes the block-wise "instantaneous" error will outperform a fixed $\epsilon$ that minimizes the average MSE. In fact, a training-based $\epsilon$ may not be optimal for *any* image, even one from the training set. In [11] (reported to outperform the authors' previous work in [12]), a block-adaptive value of $\epsilon$ is computed as the standard deviation of the block normalized by its mean. However, no justification is given for this choice. Watabe et al. propose that the $\epsilon$-filter be preceded by a "component separating" [15] filter (a low-pass filter is suggested in [15]), that aims at separating the "significant" components from the rest of the image, and preserving them from any processing. The authors of [15] indicate that this achieves a de-sensitization of the system to the value of $\epsilon$: since noise is supposedly affecting the "less significant" content, some error in the value of $\epsilon$ can be tolerated; the result would still be overall visually pleasing. However, the significant components can well contain significant amounts of MN, which are not denoised by this method [15]. In what follows, we refer to this method as CSFEF.

## 3. PROPOSED RECURSIVE $\epsilon$-FILTERING

This section presents a compression-blind recurisve $\epsilon$-filter (REF) that reduces MN noise in a video sequence by recursively applying an $\epsilon$-filter to each video frame. The $\epsilon$ of the filter is adaptively selected based on the local statistics of the image. Details of the $\epsilon$ selection and a description of the proposed REF are presented in Sections 3.1 and 3.2, respectively. First, we start by defining $\epsilon$-filters.

Consider an unknown signal $L[u]$ distorted by additive random noise $n[u]$ in the range $-J, ..., 0, ..., J$, where $u \in Z^2$ is the location of the sample in the image:

$$f[u] = L[u] + n[u], n \in -J, ..., J \tag{1}$$

Let $W(u)$ be a fixed-shape and -size window centered around $f[u]$. The output $\mathbf{y}[u]$ of the $\epsilon$-*filter with parameter* $\epsilon$, $EF(\epsilon)$, at $u$, is defined as:

$$y[u] = \frac{\sum_{v \in S(u,\epsilon)} f[v]}{|S(u, \epsilon)|} \tag{2}$$

where $S(u, \epsilon)$ is the set of pixels in $W(u)$ that differ from $f[u]$ by at most an $\epsilon$; i.e.,

$$S(u, \epsilon) = \{v \in W(u) : |f[u] - f[v]| \leq \epsilon\}$$

In words, the output of the filter at any given pixel $u$ is the arithmetic average of the values in a window centered at $u$, differing from $f[u]$ by at most $\epsilon$.

The motivation behind the EF is that MN consists of a fluctuating cloud of pixels. The cloud's fluctuation has a small amplitude, and is usually centered around 0. Averaging these fluctuations out would restore the clean values. This commonly held reasoning is correct, *under the condition that the clean values of pixels included in the filter's support are equal*. Thus, a "good" $\epsilon$ is one which guarantees that only pixels whose clean value equals the clean value of the currently filtered pixel are included in the support: $S(u, \epsilon_{good}) = \{v \in W(u) : L[u] = L[v]\}$. This is tackled in the next section.

### 3.1. Selection of $\epsilon$

Given the above motivation, we will derive the value of $\epsilon$ that minimizes the number of outliers in the filter's support. Here, 'outliers' refers to pixels whose clean value differs from the clean value of the currently filtered pixel. Therefore, assume pixel $f[u]$ is being filtered, with $f[u] = a + n[u]$. Now, $\epsilon$ should be chosen so as to maximize the probability of $f[v] \in W(u)$ be added to $S(u)$ given that $L[v] = a$. Formally,

$$\begin{aligned}
\epsilon^*(u) &= \text{argmax}_\epsilon Pr[|f[u] - f[v]| \leq \epsilon | L[v] = a] \\
&= \text{argmax}_\epsilon \frac{Pr[|f[u] - f[v]| \leq \epsilon, L[v] = a]}{Pr[L[v] = a]} \\
&= \text{argmax}_\epsilon Pr[|n[u] - n[v]| \leq \epsilon]
\end{aligned}$$

Since $n[u] \in \{-J, \ldots, J\}$, $n[u] - n[v] \in \{-2J, \ldots, 2J\}$, one solution to the above problem is

$$\epsilon^* = 2J \tag{3}$$

achieving a maximum probability of 1. Since $J$ is, in general, unknown, and varies between clouds of MN, it is approximated by the noise standard deviation $\sigma_n$: for a uni-modal noise density like the Generalized Gaussian, most of the mass will be within one standard deviation. Hence $\epsilon^* \approx 2\sigma_n$. The noise standard deviation will be approximated by the sample variance $\sigma[u]$ around $u$, which is a valid approximation in near-flat areas where MN is most annoying. Of course, a more sophisticated variance estimator could be used.

### 3.2. Recursive $\epsilon$-filter and algorithm flow

It has been pointed that the $\epsilon$-filter's performance depends on the inclusion of like pixels in its support. Because the $\epsilon$-filter operates as a low-pass filter, its output has less variation than its input. So, we propose a recursive $\epsilon$-filter (REF) scheme in which the filter input is recursively updated to include not only the initial noisy pixels, but

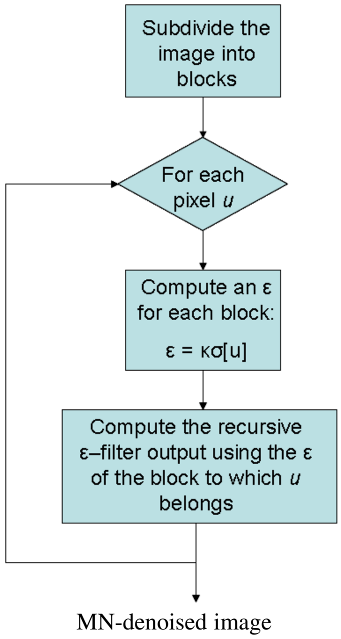



**Fig. 1**. Flowchart of proposed recursive $\epsilon$-filtering (REF) scheme.

also the *filtered* pixels as they become available. In other words, the input image to the filter is always the most recently filtered version.

Fig. 1 summarizes the steps of the algorithm: the image is divided into blocks, and the standard deviation in each is measured. $\epsilon$ for that block is selected as $\epsilon = \kappa\sigma[u]$, with $\kappa = 2$. The edge pixels are detected (using a standard Canny detector) and not filtered. Not filtering the edge pixels serves to avoid excessive blurring of visually important features. Then the block -minus its edge pixels- is filtered in a recursive fashion as described above.

## 4. EXPERIMENTAL RESULTS

This section presents sample MN-denoising results using the proposed REF scheme, and comparisons with exisiting MN-denoisers. Fig. 2 shows the denoising results on the MPEG-4 compressed Frame $10,956$ (Fig. 2(a)) of the $240\times360$ Near Science video sequence. This Near Science sequence is compressed using MPEG-4 at 4.9 Mbps, and can be downloaded from http://pdos.csail.mit.edu/scigen/#talks (the low quality version was used). Fig. 2(e) shows the obtained MN-denoised frame using the proposed compression-blind REF scheme. For comparison, Figs. 2(b), (c) & (d) show the resulting denoised frame using the existing MNR [6], CSFEF [15] and LOWTEM [7] MN-denoising schemes, respectively. It can be seen that the proposed REF MN-denoising scheme is capable of significantly reducing the mosquito noise (Fig. 2(e)) as compared to the existing MNR [6] (Fig. 2(b)) and CSFEF [15] (Fig. 2(c)) schemes. The existing LOWTEM [7] MN-denoising scheme (Fig. 2(d)) results in a significant blurring of the denoised frame as compared to the proposed REF MN-denoising scheme (Fig. 2(e)). In addition, subjective quality assessment was conducted for several MN-denoised video sequences. These assessment results also confirm that the proposed REF MN-denoising scheme is superior, as compared to the existing MN-denoising methods [6, 15, 7], in terms of impairement visibility and overall visual quality.

## 5. REFERENCES


[1] Y. Yang and N. Galatsano, “Removal of compression artifacts using projections onto convex sets and line process modeling,” *IEEE Transactions on Image Processing*, vol. 6, pp. 1345–1357, Oct. 1997.

[2] Y. Yang, N. Galatsano, and A. Katsaggelos, “Regularized reconstruction to reduce blocking artifacts of block discrete cosine transform compressed images,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, no. 6, pp. 421–432, Dec. 1993.

[3] C. A. Segall and A. K. Katsaggelos, “Pre- and post-processing algorithms for compressed video enhancement,” in *IEEE Asilomar Conference on Signals, Systems, and Computers*, 2000, pp. 1369–1373.

[4] A. Jacquin, H. Okada, and P. Crouch, “Content-adaptive postfiltering for very low bit rate video,” in *IEEE Data Compression Conference*, 1997, pp. 111–120.

[5] C. Fenimore, J. Libert, and P. Roitman. (2000) Mosquito noise in MPEG-compressed video: test patterns and metrics. [Online]. Available: http://www.itl.nist.gov/div895/docs/MosquitoNoise2000.pdf

[6] S. D. Corso, C. Miro, and J. Jung, “MNR: A novel approach to correct MPEG temporal distortions,” *IEEE Transactions on Consumer Electronics*, vol. 49, no. 1, pp. 229–236, Feb. 2003.

[7] F.-X. Coudoux, M. G. Gazalet, and P. Corlay, “A postprocessor for reducing temporal busyness in low-bit-rate video applications,” *Signal Proc.: Image Comm.*, vol. 18, no. 1, pp. 455–63, July 2003.

[8] J. Apostolopoulos and N. Jayant, “Postprocessing for very low bit-rate video compression,” *IEEE Transactions on Image Processing*, vol. 8, no. 8, pp. 1125–9, Aug. 1999.

[9] H. Harashima, K. Arakawa, and T. Fujii, “Nonlinear digital filter in signal processing (japanese),” *Trans. IEICE A, J-65-A*, vol. 18, no. 4, pp. 297–304, Apr. 1982.

[10] H. Hisamitu and K. Arakawa, “Time and spatial $\epsilon$-filter for improving image quality of compressed video,” in *IEEE Midwest Symposium on Circuits and Systems*, 2004.

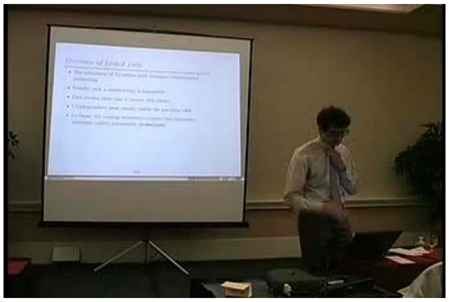
[11] T. Shohdohji, “Method and apparatus for the reduction of mosquito noise in decoded images,” Patent 20 050 117 807, 2005.

[12] T. Shohdohji, Y. Sasaki, and Y. Hoshino, “An efficient approach to the reduction of mosquito noise for the JPEG/JPEG2000 decoded image by using epsilon filter,” in *IST’s NIP18: 2002 Int’l Conference on Digital Printing Technologies*, 2002.

[13] K. Arakawa, “Fuzzy rule-based signal processing and its application to image restoration,” *IEEE JSAC*, vol. 12, no. 9, Dec. 1984.

[14] B. Ramamurthi and A. Gersho, “Non-linear space-variant processing for block coded images,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-34, no. 5, Oct. 1986.

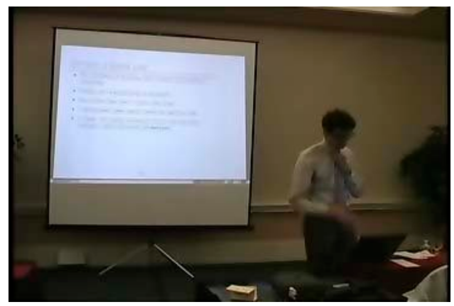
[15] H. Watabe, Y. Arakawa, and K. Arakawa, “Nonlinear filters for multimedia applications,” in *IEEE Int. Conf. Image Processing*, 1999, pp. 174–178.



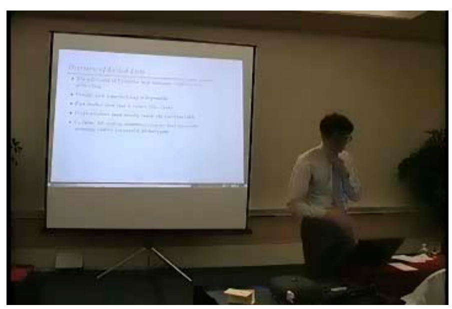
(a) Noisy

(b) MNR [6]

(c) CSFEF [15]



(d) LOWTEM [7]



(e) Proposed REF

**Fig. 2**. Results of proposed and existing MN-denoising schemes on Frame $10,956$ of the $240\times360$ Near Science video sequence.