Improved EMVD for RAW video denoising

Zhe Zheng, Gabriele Facciolo, Pablo Arias

Université Paris-Saclay, CNRS, ENS Paris-Saclay, Centre Borelli, 91190 Gif-sur-Yvette, France
{zhe.zheng,gabriele.facciolo,pablo.arias-martinez}@ens-paris-saclay.fr

Résumé – Ces derniers temps, le domaine du débruitage d'images et de vidéos connaît une révolution grâce à l'apprentissage profond. Ces méthodes sont plus performantes que les approches traditionnelles basées sur des modèles dans presque tous les problèmes de restauration d'images et de vidéos. Dans cet article, nous nous concentrons sur une approche récurrente récemment proposée pour le débruitage vidéo, à savoir la méthode de débruitage vidéo multi-étapes efficace (EMVD) [1]. Elle se compose de trois étapes, à savoir la fusion, le débruitage et le raffinement. Nous proposons d'améliorer les performances de trois manières. (1) Nous appliquons la compensation de mouvement pour mieux utiliser la redondance temporelle, (2) nous appliquons la stabilisation de la variance pour aider ce réseau léger à gérer le bruit dépendant du signal et (3) nous découplons la détection des occultations et la prédiction des poids de fusion.

Abstract – Recently, the field of image and video denoising is undergoing a revolution thanks to deep learning. These methods outperform traditional model-based approaches in almost every image/video restoration problem. In this paper, we focus on a recently proposed recurrent approach for video denoising, namely Efficient Multi-stage Video Denoising method (EMVD) [1]. It consists of three stages, including fusion, denoising, and refinement stages. We propose to improve performance in three ways. (1) We apply motion compensation to make better use of temporal redundancy, (2) we apply variance stabilization to help this light-weighted network deal with signal-dependent noise and (3) we decouple occlusion detection and fusion weights prediction.

1 Introduction

Digital images/videos are inevitably corrupted by the nature of imaging process, like the photon counting or electronic readout noise. These result in noise which reduces the visual quality and limits further usage of images and videos. As a consequence, it is highly needed to perform restoration techniques to obtain high-quality images/videos. Denoising is one of the most important steps among the camera pipeline. In the past decades many methods have been proposed for this purpose. Recently, along with the development of the computational power of GPUs, models based on deep neural networks, drew considerable attention since they do show great performance on this task.

In the context of video denoising, making use of temporal redundancy is of critical importance. This characteristic of videos should facilitate denoising performance compared with single image denoising, as it provides additional information. Deep learning based methods have been applied to video denoising since 2016 [2], and several methods based on convolutional neural networks (CNN) have been proposed, e.g. [3, 4, 5]. In spite of their good performance, deep learning based methods have not been adopted by the industry, especially for real-time applications or low-power devices, due to their high computational cost. Efficiency is a key aspect for practical video restoration. Yet, the methods in most of the current literature focus on quality rather than efficiency. For example, they process many input frames (past and future) to produce each output frame. On the other hand, traditional approaches for real-time applications often rely on *recurrence*: a recurrent method receives as input its own output for the previous frame or more generally, an encoding of it. This allows to incorporate information from past frames, without incurring in excessive cost [6, 7]. This can reduce computational and memory cost, plus it enforces temporal consistency in a natural and direct way.

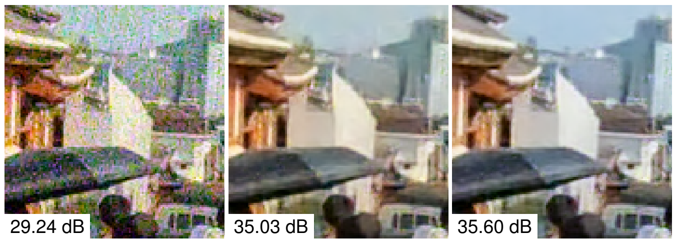

Figure 1: Comparison between results given by using warping or not, PSNR values are indicated. Left: noisy frame; Center: EMVD result without warping; Right: improved result using warping.

Recurrent CNNs for video processing have attracted attention [8], and in this paper we will focus on the EMVD (Efficient Multi-stage Video Denoising) method introduced in [1]. It is a lightweight model which combines temporal averaging with spatial denoising. It has a very low computational cost, interpretable architecture and still reports a performance comparable to other state-of-the-art methods.

In this work we propose three modifications to the EMVD method. (1) We apply motion compensation to make better use of temporal redundancy; (2) we apply variance stabilization to help this lightweight network deal with signal-dependent noise and (3) we set the variance of temporal fusion as an additional input to fusion network. We evaluate the obtained results on a synthetic dataset of RAW videos. Ablation studies are also reported, demonstrating a significant performance improvement due to the motion compensation.

2 Review of EMVD

The EMVD method is composed of 3 convolutional networks, which correspond to 3 processing stages: fusion, pre-denoising and refinement. The input data is mapped into a transformed domain by a color and a frequency trainable transforms. We will denote the clean and noisy video by $y_t(x)$ and $z_t(x)$ respectively, where x represents spatial location on the frame and t is the frame index.

Trainable transforms. Each noisy frame z_t is pre-processed by linear trainable color and frequency transforms. The color transform $\mathcal{T}_c$ is a matrix $M \in \mathbb{R}^{C\times C}$, where C is the number of channels (4 for a packed RAW frame). It aims at decorrelating the color to luminance-chrominance representation, and its initial value is the same as proposed in [9]. The frequency transform decorrelates the input frequencies into four half-resolution components: one low-pass LL subband and three high-pass LH, HL, HH subbands. It is initialized with the Haar transform, whose decomposition and reconstruction filters are denoted by ψ and ϕ. At the output of the network the inverse transforms are applied. To enforce the invertibility of the transforms during training, two additional loss terms are added:

$$\mathcal{L}_c = \|MM^T - I_C\|_F^2, \quad \mathcal{L}_f = \|\psi\phi^T - I_2\|_F^2. \tag{1}$$

Temporal fusion. EMVD achieves temporal denoising by keeping running frame average $\bar{y}_t$, and its corresponding variance map $\bar{\sigma}^2$:

$$\bar{y}_t = \bar{y}_{t-1} \odot \gamma_t + z_t \odot (1-\gamma_t), \tag{2}$$

$$\bar{\sigma}_t^2 = \bar{\sigma}_{t-1}^2 \odot \gamma_t^2 + \sigma_t^2 \odot (1-\gamma_t)^2, \tag{3}$$

where $\odot$ denotes the element-wise product. The average weight maps γ_t are between zero and one, and are computed by the fusion network FCNN, which takes as input the absolute value of the difference between the LL subbands of z_t and the previous average $\bar{y}_{t-1}$, together with the input's noise variance map σ_t^2:

$$\gamma_t = \text{FCNN}(|z_{LL,t} - \bar{y}_{LL,t-1}|, \sigma_t^2). \tag{4}$$

The fusion weights should allow temporal averaging at locations where z_t and $\bar{y}_{t-1}$ are well aligned and prevent averaging if changes are detected. In these locations, the output should coincide with the noisy input z_t. The running average is initialized with the first noisy frame.

Denoising and refinement. The remaining stages of EMVD apply a spatial denoising to $\bar{y}_t$. This is divided into two steps: a pre-denoising and a final refinement. The pre-denoising is given by a denoising network DCNN, which takes as input the temporal average and its variance map, plus the LL subband of the noisy frame

$$\tilde{y}_t = \text{DCNN}(\bar{y}_t, z_{LL,t}, \bar{\sigma}_t^2). \tag{5}$$

The refinement is a weighted average of the pre-denoised frame $\tilde{y}_t$ and the temporal average, with weights computed by the refinement network RCNN:

$$\omega_t = \text{RCNN}(\tilde{y}_t, \bar{y}_t, \bar{\sigma}_t^2) \tag{6}$$

$$\hat{y}_t = \bar{y}_t \odot \omega_t + \tilde{y}_t \odot (1-\omega_t). \tag{7}$$

The EMVD method can be easily extended to a multiscale structure by applying frequency transform several times. In our experience, the effect of this multiscale structure is marginal. For the sake of simplicity, we omit it in this work. Please refer to [1] for details.

3 Improvements on EMVD

Motion compensation. Since the temporal fusion (2) is just a temporal average, a good frame alignment is essential to attain good results. This is especially true for videos with background movement. In our experiment, we use the REDS dataset [10], which is captured by a hand-held camera, resulting in a lot of motion in these sequences. We choose the $TV-\ell_1$ method [11, 12] to compute the optical flow between the noisy frames t and $t-1$, which we use to warp the previous temporal average $\bar{y}_{t-1}$ and align it to z_t. This will enable us to make better use of temporal redundancy.

Variance stabilization. The noise in the RAW video can be approximated as a heteroscedastic Gaussian with signal-dependent variance. A variance stabilizing transform (VST) can transform this noise to a standard additive Gaussian white noise (AGWN). VSTs are a common way to adapt a AGWN denoiser to real noise. With learning-based methods, VSTs are not necessary anymore, as it has been shown that the networks can handle signal dependent noise. However, since the networks in EMVD are rather small, stabilizing the variance might help. The procedure of applying VST involves three steps. First, we transform the noisy data by a nonlinear VST which is designed for a specific noise model. Then, we denoise such transformed data. Lastly, we apply an inverse VST to the denoised data, obtaining data in the original range.

For the signal-dependent Gaussian noise we assume $n_t(x) \sim \mathcal{N}(0, ay_t(x)+b)$, and we use the following VST: $f(x) = \frac{2}{a}\sqrt{ax+b}$. After the VST, we have approximately AWGN with unit variance. We invert the VST with the algebraic inverse $f^{-1}(x) = \frac{ax^2}{4} - \frac{b}{a}$. This inverse is known to introduce a bias for dark values [13], but since the bias is deterministic, the network can learn to compensate it.

Decoupling occlusion detection and fusion weights prediction. In the original EMVD paper [1], the fusion network FCNN aims at estimating the combination weights between temporal fusion $\bar{y}_{t-1}$ and the current noisy frame z_t. These weights fulfill two goals: (1) They have to avoid using the motion compensated $\bar{y}_{t-1}$ in locations where the alignment failed, and (2) in the regions where the alignment is accurate, the weight has to be chosen to determine an optimal temporal fusion. For the latter, it is reasonable to give as input to the fusion network both the estimated variances of the current noisy frame σ_t^2 and of temporal fusion $\bar{\sigma}_{t-1}^2$. Indeed, assuming that the alignment is correct at pixel x, the optimal weights (in the means square error sense) are given by the variance ratio (VR) $\mathbb{V}(z_t(x))/(\mathbb{V}(\bar{y}_{t-1}(x)) + \mathbb{V}(z_t(x)))$. Note however, that in [1], only the variance of z_t is given as input to the network (see Eq. (4)).

Directly adding the estimated temporal fusion variance as input to the fusion network is problematic. Indeed, $\bar{\sigma}_{t-1}$ itself is a function of the previous fusion weights γ_{t-1}. This feedback loop creates a bias in the fusion network towards low fusion weights. So we propose to decouple the two tasks of the fusion network. We first compute the alignment weights as in [1], and then we multiply them by the optimal fusion weights:

$$\gamma_t^{occ} = \text{FCNN}(|z_{LL|t} - \bar{y}_{LL|t-1}|, \hat{\sigma}_t^2), \tag{8}$$

$$\gamma_t = \gamma_t^{occ} \times \frac{\sigma_t^2}{\bar{\sigma}_{t-1}^2 + \sigma_t^2}. \tag{9}$$

4 Experiments

Training settings. At the beginning of each epoch we load a set of spatio-temporal volumes. We sample crops of a spatial size 136×136 with 5 consecutive frames and choose mini-batches of size 2 to train the network. By choosing 5 consecutive frames, the recurrence is run 4 times (the first frame is used as initialization for the temporal fusion). Following the terminology used in recurrent networks, we call "unrollings" each iteration of the network.

The total loss is the sum of the losses for the color and frequency transforms (1), and ℓ_1 image reconstruction loss as criterion in the training. The reconstruction loss is a weighted sum of the outputs of the unrollings:

$$\text{loss}(\hat{y}_t, \tilde{y}_t) = \sum_{t=1}^{n} w_t \left(\lambda_o \|y_t - \hat{y}_t\|_1 + \lambda_d \|y_t - \tilde{y}_t)\|_1 \right), \tag{10}$$

where $\sum_{t=1}^{n} w_t = 1$. Note that we have added a loss term for the output of the pre-denoising network $\tilde{y}_t$, which is absent in the original work [1]. The reason for this is that there are several combinations of refinement weights w_t and pre-denoising $\tilde{y}_t$ which verify $y_t \approx \bar{y}_t \odot w_t + \tilde{y}_t \odot (1 - w_t)$. In most cases, the training converges to the expected solution, where $\tilde{y}_t$ is a preliminary estimation of the clean frame y_t (typically over-smoothed), and is then refined by adding back some details from $\bar{y}_t$. But in some cases the training converges to other solutions which are less interpretable, although they might have a similar final PSNR. We use $\lambda_o = 100$ and $\lambda_d = 1$, and in this way we ensure convergence to the expected solution.

To speed up the training, we use a transition strategy for the unrolling weights w_t. We set 100% weight on the first unrolling in the first 20 epochs, in this way we only need to run the first unrolling. In the next 5 epochs, we gradually shift the weights to the last unrolling. The final weights are, 90% on the last unrolling, and the remaining 10% weights divided evenly among the first 3 unrollings. The reason why we prefer putting most of the weight on the last unrolling is that, at test time the network will be applied to long sequences of frames. Although the network was not trained with many unrolled frames the last unrolling is assumed to be the closest to the steady state in the testing.

We use the Adam optimizer [14] to update the weights. The total training takes 100 epochs. During the first 70 epochs, the network is trained with a fixed learning rate (2e-4 in our experiments) and in the following 30 epochs, the learning rate would be reduced at each epoch linearly to 0.

Dataset. We use data from the public REDS [10] dataset. The original videos are RGB captured at 30 frames per second. This dataset consists of 270 sequences with 90 frames of size 1280×720 pixels, split into 240 sequences for training and 30 of them for testing and validation. To simulate RAW data, we use an unprocessing pipeline similar to that of [15]. This unprocessing consist in inverting the main steps of an image camera pipeline: inverse tone mapping; inverse Gamma curve; inverse color matrix; inverse white balance; mosaicking.

After unprocessing, we added noise of two levels to the RAW sequences, using noise model which are estimated from the CRVD dataset [16], ISO3200 and ISO12800. We store each image as 4 channel images of half the size. Each channel corresponds to a phase of the mosaicking pattern.

Experimental results. We report the ablation study on the methods we proposed in Table 1. To assess how the network capacity would influence the performance, we consider the EMVD under two configurations, one allocating a small capacity with 3 layers and 16 features for each one of the three networks, the other one with a big capacity has 5 layers and 64 features for all three networks. We use "S" and "L" for small and big configurations respectively in the table. In all cases, warping leads to big improvement. This is reasonable, since there is a lot of motions in the REDS dataset. Without temporal alignment, the fusion weights are mostly 0, meaning that the network just performs a single image denoising (see Figure 1).

Table 1: Ablation study of proposed improvements in terms of PSNR. Results for two ISOs, and two network capacities.

ISO	net	full	no VST	no VR	no VST, no VR	no warp
12.8k	S	36.21	36.32	36.25	36.24	35.66
	L	37.74	37.77	37.78	37.84	37.09
3.2k	S	40.95	40.77	40.77	40.86	37.37
	L	42.12	42.04	42.12	42.02	41.63

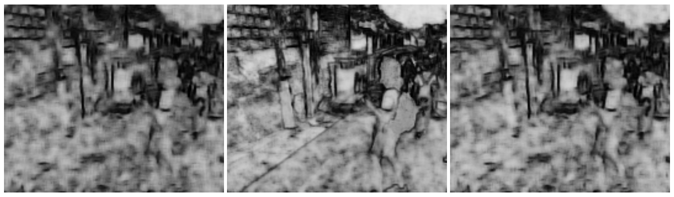
Figure 2: Comparison between fusion weights under different settings. From left to right: no VR, no VST, full.

For the other modifications, i.e. VST and the multiplication by the variance ratio (VR), we obtain very similar PSNR values. Inspecting the intermediate results we observed that the VST does not improve them either, while exploiting extra variance information can lead to 0.2-0.4 dB gains in the output of temporal fusion, depending on the capacity of networks. These modifications are expected to improve temporal fusion. Figure 2 shows different fusion weights under different settings. We can see that without the VST, the fusion weights are more correlated with the image content. This is to be expected, as the variance is proportional to the image. It can also be observed that when adding the product with the variance ratio, the fusion weights tend to be higher (e.g. in the sky), which implies more temporal averaging.

5 Conclusions

In this paper, we reviewed the EMVD method, a lightweight and interpretable recurrent video denoising CNN. We proposed to improve this method in three ways, including applying VST, incorporating the variance of temporal fusion in the computation of the fusion weights, and motion compensation. Among these, the latter makes significant improvement, which stresses the importance of using temporal redundancy in video processing. The other modifications have a positive effect on the temporal fusion (specially the VR), however this gain does not reflect in the final PSNR. A possible reason for this is that the gains are located in smooth parts of the image that are easy to handle by denoising network.

References

[1] M. Maggioni, Y. Huang, C. Li, S. Xiao, Z. Fu, and F. Song, "Efficient multi-stage video denoising with recurrent spatio-temporal fusion," in *CVPR*, 2021.

[2] X. Chen, L. Song, and X. Yang, "Deep rnns for video denoising," in *Applications of digital image processing XXXIX*. SPIE, 2016, vol. 9971, pp. 573–582.

[3] A. Davy, T. Ehret, J.-M. Morel, P. Arias, and G. Facciolo, "A non-local cnn for video denoising," in *ICIP*. IEEE, 2019.

[4] M. Claus and J. van Gemert, "Videnn: Deep blind video denoising," in *CVPR*, 2019.

[5] M. Tassano, J. Delon, and T. Veit, "Fastdvdnet: Towards real-time deep video denoising without flow estimation," in *CVPR*, 2020.

[6] P. Arias and J.-M. Morel, "Kalman filtering of patches for frame-recursive video denoising," in *CVPR Workshops*, 2019.

[7] T. Ehret, J.-M. Morel, and P. Arias, "Non-local kalman: A recursive video denoising algorithm," in *ICIP*. IEEE, 2018.

[8] T. Isobe, X. Jia, S. Gu, S. Li, S. Wang, and Q. Tian, "Video super-resolution with recurrent structure-detail network," in *ECCV*. Springer, 2020, pp. 645–660.

[9] A. Buades and J. Duran, "Cfa video denoising and demosaicking chain via spatio-temporal patch-based filtering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 11, pp. 4143–4157, 2019.

[10] S. Nah, S. Baik, S. Hong, G. Moon, S. Son, R. Timofte, and K. Mu Lee, "Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study," in *CVPR Workshops*, 2019.

[11] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime tv-l 1 optical flow," in *Joint pattern recognition symposium*. Springer, 2007.

[12] J. S. Pérez, E. Meinhardt-Llopis, and G. Facciolo, "Tv-l1 optical flow estimation," *Image Processing On Line*, vol. 2013, pp. 137–150, 2013.

[13] M. Makitalo and A. Foi, "Optimal inversion of the generalized anscombe transformation for poisson-gaussian noise," *IEEE transactions on image processing*, vol. 22, no. 1, pp. 91–103, 2012.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[15] T. Brooks, B. Mildenhall, T. Xue, J. Chen, D. Sharlet, and J. T. Barron, "Unprocessing images for learned raw denoising," in *CVPR*, 2019.

[16] H. Yue, C. Cao, L. Liao, R. Chu, and J. Yang, "Supervised raw video denoising with a benchmark dataset on dynamic scenes," in *CVPR*, 2020.