Video joint denoising and demosaicing with recurrent CNNs

Valéry Dewil† Adrien Courtois† Mariano Rodríguez† Thibaud Ehret†
Nicola Brandonisio* Denis Bujoreanu* Gabriele Facciolo† Pablo Arias†

† Université Paris-Saclay, CNRS, ENS Paris-Saclay, Centre Borelli, 91190, Gif-sur-Yvette, France
* Huawei Technologies France SASU

https://centreborelli.github.io/RVDD/

Abstract

Denoising and demosaicing are two critical components of the image/video processing pipeline. While historically these two tasks have mainly been considered separately, current neural network approaches allow to obtain state-of-the-art results by treating them jointly. However, most existing research focuses in single image or burst joint denoising and demosaicing (JDD). Although related to burst JDD, video JDD deserves its own treatment. In this work we present an empirical exploration of different design aspects of video joint denoising and demosaicing using neural networks. We compare recurrent and non-recurrent approaches and explore aspects such as type of propagated information in recurrent networks, motion compensation, video stabilization, and network architecture. We found that recurrent networks with motion compensation achieve best results. Our work should serve as a strong baseline for future research in video JDD.

1. Introduction

Every optical camera, from mobile phones to professional DSLRs, uses an image signal processor (ISP) which aims at producing good quality sRGB images from the raw input captured by the sensor. ISPs implement numerous operations, some of which can be quite complex. A considerable effort goes into designing, implementing and tuning the image processing pipeline to achieve the best possible picture quality using limited computational resources.

Two important components of a camera pipeline are denoising and demosaicing. They are typically applied separately: first a denoising method is applied on the raw data and then the denoised raw is demosaiced [46, 67, 45, 31]. The main benefit of this approach is that denoising is applied on one third of the data of the RGB image. Recent works have proposed to invert the order of these operations in order to better preserve the small image structures at the denoising stage. Demosaicing before denoising produces correlated noise, however it is shown in [28] that denoisers can be adapted to handle this correlated noise yielding results that surpass the ones of denoising before demosaicing.

Yet, the ideal situation is to combine these two steps into a single joint denoising and demosaicing module. Not only this should lead to better results but it would also simplify the camera pipeline by combining two deeply interconnected modules into a single one.

Several methods have been proposed for joint denoising and demosaicing, from traditional model-based methods [32, 6, 18, 23, 37] to more recent data-driven approaches [17, 10, 55, 35, 11]. However, most of works focus on single images [23, 32, 22, 34, 17, 25, 37, 62] or bursts [35, 11, 19, 21], while the case of video has received little attention so far. Early video demosaicing works assume that the raw is noiseless [61, 39]. Patch-based methods have been proposed in [66, 5] but treat the denoising and demosaicing separately. In [9] an image demosaicing algorithm is applied to the noisy raw frames, which are then denoised by a self-supervised video denoising network.

There are obvious similarities between bursts and videos. In both cases the focus is to use multiple frames as input. Temporal aggregation of information should benefit both denoising and demosaicing. Indeed, when multiple input frames are available missing values on the current frame can be observed in neighboring frames. This is the approach taken by [14, 60], which obtains a super-resolved sRGB image exploiting the hand-held camera motion. Several learning based approaches have been proposed for burst JDD either with supervised [35, 19, 20, 21] or self-supervised [11] training. Very recently some authors have attacked the problem using neural fields [47, 41]. A related problem is raw burst super-resolution, where the goal is to obtain a super-resolved sRGB image [60, 3, 36, 2].

In spite of the similarities between burst and video JDD there are important differences. Since the objective of burst processing is to produce a single image, many frames are

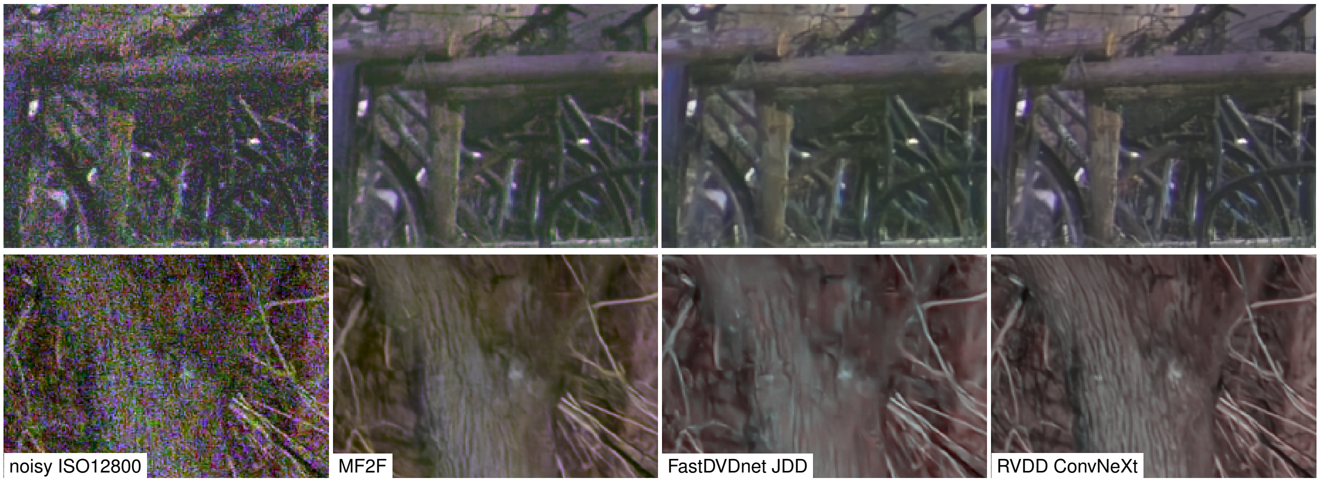

Figure 1: Results obtained with our joint denoising and demosaicing method (RVDD) on real raw videos from the CRVD dataset [64]. For comparison we show results obtained with the self-supervised video denoising method MF2F [9] and with an adaptation of FastDVDnet [57] to JDD

usually processed/aggregated. In contrast, a realistic video processing ISP cannot afford to maintain a rolling window containing dozens of frames. Moreover, the processed video needs to be temporally consistent. These constraints shape already the very few methods dedicated to raw video denoising, which either resort to recurrent techniques [13, 1, 40, 24, 44], or limit themselves to small temporal windows of a few frames [57, 64, 53, 63, 59, 7, 54, 33].

Although there is a large body of work in related problems, the problem of video JDD, to the best of our knowledge, has not yet been addressed with learning based approaches in spite of it being a basic operation which is part of every sRGB video acquisition pipeline. Our goal in this work is to set a CNN baseline on the problem of video JDD.

Contributions. In this work we tackle the problem of raw video JDD using neural networks. Our contributions are:
(1) We propose a recurrent CNN for video JDD. We provide extensive ablations considering recurrent and non-recurrent versions, with and without explicit motion compensation, among others. Our results confirm that a simple early fusion architecture with motion compensation and recurrence is a strong baseline for video JDD.

(2) For quantitative evaluation and training, we provide a simulated raw-to-sRGB realistic dataset (based on REDS [42]). Our dataset is tailored to the characteristics of CRVD [64] (a public real raw video dataset). In this way we can apply the trained networks on the real CRVD dataset (see Figure 1). We consider two versions of our dataset: with and without motion stabilization. This allows to evaluate the generalization of JDD networks across datasets with different motion statistics.

Our dataset, code and results are available at the project's web page[1] and could serve as a baseline for future publications on the subject.

2. Recurrent CNN for video JDD

We denote by f a noisy raw video of size $W \times H$, and by f_t with $t = 1, \dots, T$ one of its frames. The video f is a mosaiced noisy version of the linear RGB video u ($W \times H \times 3$). We denote by M the mosaicing operator, and $u_t^{\mathrm{M}} = Mu_t$ the clean raw frame. We assume the widely used heteroscedastic Gaussian approximation of the real sensor noise [15]:

$$f_t = u_t^{\mathrm{M}} + n_t \odot \sqrt{au_t^{\mathrm{M}} + b} \quad \text{with} \quad n_t \sim \mathcal{N}(0, I), \quad (1)$$

where $\odot$ denotes the element-wise product, n_t is an image of Gaussian white noise of mean $\mu = 0$ and variance $\sigma^2 = 1$ and $a, b \geq 0$ are the parameters of the noise model. In this model, the noise is white Gaussian with a variance that depends on the clean value of the pixel. For pixel x in raw frame t the variance of the noise is $au_t^{\mathrm{M}}(x) + b$.

For a video restoration task, it is impractical to consider a large window of input frames, which makes recurrent networks an appealing choice for integrating temporal information across a larger number frames beyond the input window. Recurrent networks have been applied to video denoising [40, 24, 44] and super-resolution [50, 27, 16]. To address for the first time the video JDD problem, we define a simple architecture that combines recurrence on the output frame [50] and feature recurrence [27, 16, 24].

A diagram of the proposed Recurrent Video joint Denoising and Demosaicing (RVDD) method is given in Figure 2. We consider a standard U-Net CNN (similarly to

[1] `https://centreborelli.github.io/RVDD`

[57, 53, 44, 63]), which we denote by $\mathcal{F}$, that receives four inputs: the previous RGB output $\widehat{u}_{t-1}$, the current and next raw noisy frames f_t, f_{t+1}, and the feature map from the last hidden layer φ^L_{t-1} of the previous frame (with C channels and spatial resolution $W \times H$). The raw inputs f_t and f_{t+1} are demosaiced with the Hamilton-Adams method [30], which we denote by $\mathcal{D}$. The adjacent frames and activation maps are aligned to frame t using warping operators $\mathcal{W}_{t-1,t}$ and $\mathcal{W}_{t+1,t}$ to compensate for motion:

$$\widehat{u}_t = \mathcal{F}\left(\mathcal{W}_{t-1,t}\varphi^L_{t-1}, \mathcal{W}_{t-1,t}u_{t-1}, \dots \right. \\ \left. \mathcal{D}(f_t), \mathcal{W}_{t+1,t}\mathcal{D}(f_{t+1})\right). \quad (2)$$

The warping operator $\mathcal{W}_{t\pm1,t}$ is given by an optical flow $v_{t,t\pm1}$ from frame t to $t \pm 1$:

$$\mathcal{W}_{t\pm1,t}u_{t\pm1}(x) = u_{t\pm1}(x + v_{t,t\pm1}(x)). \quad (3)$$

We interpolate the warped frame with a differentiable version of bicubic interpolation so as to be able to backpropagate gradients during training.

Optical flows are estimated on the noisy raw video. The raw frames are downsampled to half resolution via average pooling (the 4 pixel values in each Bayer cell are averaged). We use TV-L1 [65] and upscale the result to the full resolution. By operating the optical flow at half resolution we reduce the computational time and the noise level.

The image inputs $\mathcal{W}_{t-1,t}u_{t-1}$, $\mathcal{D}(f_t)$, and $\mathcal{W}_{t+1,t}\mathcal{D}(f_{t+1})$ are concatenated along the channel dimension into a tensor of size $W \times H \times 9$. The feature map input $\mathcal{W}_{t-1,t}\varphi^L_{t-1}$ is concatenated to the feature map of the first hidden layer φ^1_t resulting in a tensor of size $W \times H \times 2C$. Concatenating after feature extraction favors a balanced combination of the previous features with the new ones.

Basic recurrent baseline. We also consider a basic recurrent CNN, denoted as RVDD-basic, keeping the same U-Net architecture but with only two inputs: the current noisy frame f_t and the previous RGB output $\hat{u}_{t-1}$, i.e.

$$\widehat{u}_t = \mathcal{F}\left(\mathcal{W}_{t-1,t}u_{t-1}, \mathcal{D}(f_t)\right). \quad (4)$$

This will serve as a recurrent baseline in Section 6.

3. Modified FastDVDnet for JDD

FastDVDnet is a video denoising CNN introduced in [57]. It takes as input a stack of five consecutive noisy frames, and processes them with two cascaded U-Nets. The first U-Net is applied three times on each set of three contiguous frames. The three outputs are then used as input for the second U-Net that produces the final result.

We propose a simple adaptation of FastDVDnet to perform joint denoising and demosaicing. Following [28] we demosaic the frames (using the Hamilton-Adams demosaicing [30, 29]) before feeding them to FastDVDnet. The network will therefore remove the demosaic noise. This allows for a fair comparison with the networks proposed in Section 2 in the sense that the network operates at the full output resolution. Indeed, training FastDVDnet to operate on raw frames and demosaicing the result afterwards leads to substantially worse results. A different variant of FastDVDnet for JDD is discussed in the supplementary material.

4. Datasets

For a quantitative comparison we generated a synthetic dataset of raw noisy videos with clean RGB ground truth. The dataset is tailored to model the CRVD dataset [64]. The latter consists of real noisy raw videos of 50 outdoors scenes acquired with a surveillance camera at five ISO levels, and we will use them for visual evaluation on real data.

For our synthetic dataset we use sequences from the sRGB REDS-120 dataset [43], which consists of 270 dynamic sequences (split in 240 training and 30 validation sequences) of outdoors scenes taken in daylight conditions, with frame rate 120 FPS and size 1280×720. We temporally subsampled each sequence to a frame rate of 40 FPS. The subsampled sequences have 90 frames each.

The sRGB sequences are transformed to the raw domain by applying an simple inverse camera pipeline as in [4], consisting of the inverses of tone-mapping, gamma correction, color correction, white balance and mosaicing. We adapted this "unprocessing" method to the CRVD dataset. We used the CCM matrix provided by the authors of the CRVD dataset. We randomly sampled white balance coefficients as in [4] and kept them constant for all frames in a given sequence.

We then added a heteroscedastic Gaussian noise with parameters estimated from the CRVD dataset. The noise parameters were estimated using *Ponomarenko*'s noise estimation algorithm [8, 48] that estimates the noise level curve (intensity, standard-deviation) from an image. The algorithm was applied on all the frames of the CRVD dataset with a given ISO. The linear model was determined by minimizing the least-square fit on the estimated noise curves.

We generate datasets for two ISO levels out of the five in CRVD: 3200 and 12800.

Stabilized dataset. The sequences in REDS-120 were captured with a handheld camera resulting in large camera motion. While our networks rely on an external optical flow for explicit motion compensation, FastDVDnet does not. The idea is that U-Nets, with their large receptive field, should be capable –to a certain degree– of implicitly handling the motion in the sequence. In order to ease the job of FastDVDnet, we create a second version of our dataset

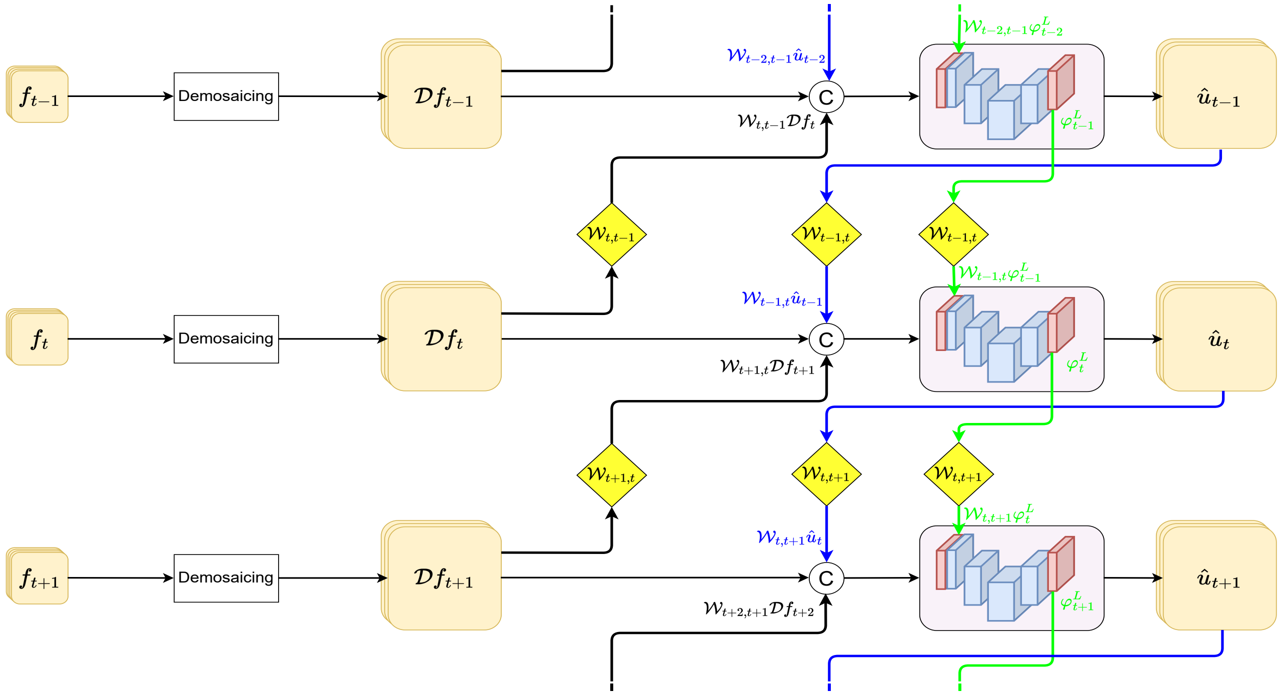

Figure 2: **Joint denoising & demosaicing** in the RGB domain. Data inputs and outputs are represented as colored rounded squares. Small squares represent the packed raw frames whereas large squares represent RGB frames.

where the motion is stabilized using an homographic offline video stabilization algorithm [51, 52], that reduces camera motion and makes it more predictable.

5. Training details

Training details. At the beginning of each epoch we load into RAM a random segment of 10 consecutive frames from each sequence in the training set, together with the optical flows, masks, etc. From these spatio-temporal volumes, we define a set of 3D crops with a stride of three pixels in all dimensions (both spatial and temporal). During the entire epoch, mini-batches are sampled at random from these set of crops. Crops have a spatial size of 272×272 with a number of frames dictated by the network and the number of unrollings (e.g. for training 4 unrollings we need 5 consecutive frames for the recurrent JDD network, and 6 if we use the future frame). The denoising network processes each 3D crop in the mini-batch and returns an output which can be (a) a single frame for the non-recurrent network or (b) $T + 2$ frames for a recurrent network trained with T unrollings (T frames, plus one additional frame for the first unrolling and one for the last if the future frame is used). We use the AdamW optimizer to update the weights with a decay parameter of 0.01. We perform 70 epochs, with a fixed learning rate and then 30 epochs reducing it at each epoch linearly to 0. We start with a learning rate of 1.6e-4.

For the recurrent networks the loss is a weighted average of the L1 losses of the outputs of the T unrollings. The weights change during training, shifting gradually from the first unrolling to the last. For more details refer to the supplementary material.

Training details for FastDVDnet We initially trained our modified architecture using the same hyperparameters (learning rate, patch size and batch size) from [57]. However, the resulting networks were unstable at test time, creating very high output values in flat regions. We fixed these issues by removing the batch normalization [26] and adapting the hyper-parameters, resulting in a patch size of 68, batch size of 2 and learning rate of 10^{-4}. The learning rate is reduced by a factor of 10 after 50 epochs; and reduced again by a factor of 100 after epoch 60. The networks are trained for 100 epochs and we keep the network with the highest validation score.

6. Experimental results

Throughout this section we use PSNR and SSIM as metrics to compare the different models. We restrict the validation dataset to the first five sequences of the simulated dataset. The networks outputs are transformed to the sRGB domain for visualization and for evaluating the PSNR/SSIM. We apply a white balance, a color matrix cor-

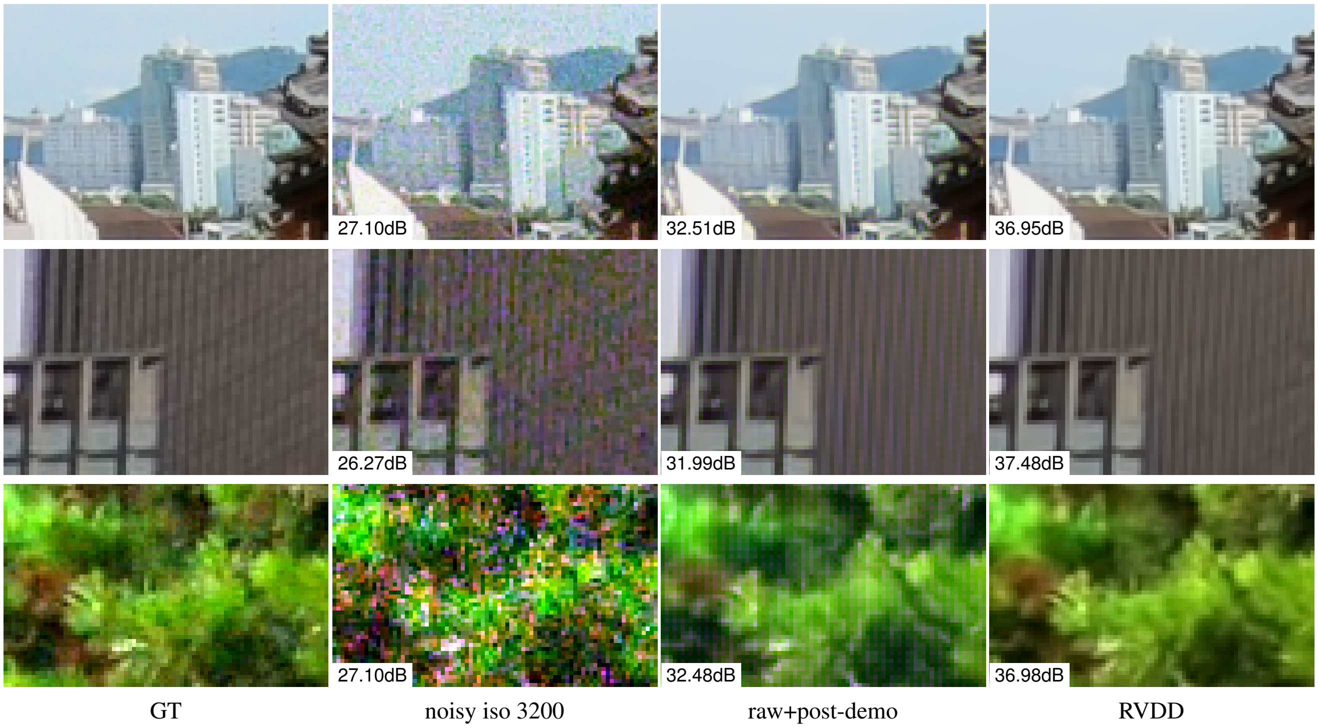

Figure 3: Comparison of our RVDD method with a raw denoiser followed by a demosaicing network [56]. Both RVDD and the raw denoising network share the same architecture. A post-processing pipeline has been applied to both results. The contrast has been enhanced in the last row. The box in the bottom-left corner contains the PSNR of the full frame.

Framework	RGB PSNR 3.2k	RGB PSNR 12.8k	raw PSNR 3.2k	raw PSNR 12.8k
RVD-basic + CDM [56]	42.54	38.95	43.38	38.96
RVDD-basic	**44.74**	**40.73**	**43.92**	**39.78**
RVDD-basic-$\mathcal{WD}$	44.59	40.59	43.80	39.67
RVDD-basic-$\mathcal{DW}$	44.36	40.33	43.56	39.39

Table 1: PSNR on the linear RGB and in the raw domain for the raw denoiser followed by a demosaicing [56] and our JDD method in the validation set of our synthetic dataset. Ignoring the pre-demosaicing in our JDD method, the architecture is the same. The results of our JDD are previously remosaiced for computing the PSNR in the raw domain. We consider two ISO levels taken from the CRVD dataset. Best results are in **bold**.

	φ^L_{t-1}	f_{t+1}	sRGB PSNR 3.2k	sRGB PSNR 12.8k	sRGB SSIM 3.2k	sRGB SSIM 12.8k
RVDD-basic	✗	✗	37.90	35.64	0.961	0.938
	✓	✗	38.12	35.72	0.962	0.941
	✗	✓	38.19	36.05	0.962	0.943
RVDD	✓	✓	**38.37**	**36.26**	**0.964**	**0.946**

Table 2: PSNR and SSIM after the pipeline (sRGB) for the different frameworks for handling the recurrence (see Section 2) in the validation set of our synthetic dataset. We consider two ISO levels taken from the CRVD dataset. Best results are in **bold**.

rection and a gamma correction. We use the inverse of the actual white balance coefficients which have been used to generate the raw dataset during unprocessing. In the supplementary material, we show PSNR/SSIM in the linear RGB domain.

JDD *vs.* raw denoising and demosaicing. We first evaluate the impact of joint denoising and demosaicing, as opposed to first denoising the raw and then demosaicing the denoised raw output. In Table 1, we compare our baseline recurrent JDD network RVDD-basic against a raw denoiser followed by a pre-trained demosaicing network [56] (we use the implementation of [12]). For the raw denoising network, we adapt the RVDD-basic network by removing the Hamilton-Adams demosaicing of the input and feeding directly the packed 4 channel raw frames. We then train it using the clean raw ground truth in the loss (instead of the linear RGB). We refer to this network as RVD-basic.

The JDD network demonstrates much better performance than first raw denoising followed by pre-trained demosaicing, even when the raw denoising network has a similar architecture than the JDD (*e.g.* same number of parameters). From an architectural point of view, the main difference is that the JDD network applies the demosaicing on

network	$\mathcal{W}$	f_{t+1}	trained on	non-stabilized				stabilized			
				sRGB PSNR		sRGB SSIM		sRGB PSNR		sRGB SSIM	
				3.2k	12.8k	3.2k	12.8k	3.2k	12.8k	3.2k	12.8k
FastDVDnet-JDD			non stab.	36.11	33.47	0.942	0.907	36.59	34.06	0.948	0.917
VDD	✗	✗	non stab.	36.42	33.89	0.945	0.913	36.71	34.26	0.949	0.921
VDD	✗	✓	non stab.	36.37	33.89	0.945	0.913	36.89	34.52	0.951	0.923
VDD	✓	✗	non stab.	37.22	34.83	0.954	0.927	37.36	34.93	0.956	0.931
VDD	✓	✓	non stab.	37.72	35.47	0.958	0.934	37.88	35.57	0.961	0.938
RVDD-basic	✓	✗	non stab.	37.90	35.64	0.961	0.938	38.08	35.78	0.963	0.942
RVDD	✓	✓	non stab.	**38.37**	**36.26**	**0.964**	**0.946**	**38.39**	**36.37**	**0.966**	**0.949**
FastDVDnet-JDD			stab.	35.53	32.76	0.937	0.897	36.92	34.57	0.952	0.924
VDD	✗	✗	stab.	36.25	33.77	0.944	0.911	37.07	34.63	0.953	0.925
VDD	✗	✓	stab.	36.16	33.57	0.944	0.908	37.22	34.65	0.954	0.926
VDD	✓	✗	stab.	37.15	34.77	0.953	0.926	37.41	34.96	0.956	0.931
VDD	✓	✓	stab.	37.66	35.42	0.958	0.934	37.94	35.65	0.961	0.939
RVDD-basic	✓	✗	stab.	37.83	35.66	0.960	0.940	38.15	35.92	0.964	0.944
RVDD	✓	✓	stab.	**38.29**	**36.22**	**0.963**	**0.945**	**38.63**	**36.50**	**0.967**	**0.950**

Table 3: PSNR and SSIM after the pipeline (sRGB) in the validation set of our synthetic dataset. We compare our JDD adaptation of FastDVDnet [57] with six variants of our network: two recurrent –RVDD-basic and the full RVDD–, and four non-recurrent networks labeled VDD: with/without warping ($\mathcal{W}$) and with/without the future frame f_{t+1}.

Architecture	sRGB PSNR		sRGB SSIM	
	3.2k	12.8k	3.2k	12.8k
RVDD-basic U-Net	37.90	35.64	0.961	0.938
RVDD-basic ConvNeXt U-Net	37.93	35.70	0.960	0.941
RVDD U-Net	38.37	36.26	**0.964**	0.946
RVDD ConvNeXt U-Net	**38.56**	**36.62**	**0.964**	**0.948**

Table 4: PSNR and SSIM after the pipeline (sRGB) for the standard U-Net and the ConvNeXt U-Net in the validation set of our synthetic dataset. We consider two ISO levels taken from the CRVD dataset. Best results are in **bold**.

the input, thus operating at the RGB resolution, whereas the raw denoising network operates in the raw domain. In particular, the JDD network outputs and propagates from frame $t-1$ to t, an RGB image $\widehat{u}_{t-1}$ which contains three times more information than the raw. To measure the impact of this aspect, we add to the comparison two degraded versions of our JDD network where only the raw frame $\widehat{u}^{\mathrm{M}}_{t-1} = M\widehat{u}_{t-1}$ is propagated. In one we mimic the temporal propagation in the raw denoising network RVD-basic, and apply the warping on the raw image

$$\widehat{u}_t = \mathcal{F}(\mathcal{D}(\mathcal{W}_{t-1,t}\widehat{u}^{\mathrm{M}}_{t-1}), \mathcal{D}(f_t)). \tag{5}$$

To warp the raw image u^{M}_{t-1} we store it in the packed raw format (i.e. as a 4 channel $W/2 \times H/2$ image where each channel contains one phase of the Bayer pattern) and warp each channel. This is not ideal, since the phases of the Bayer pattern are downsampled versions of the color channels and are heavily aliased. Therefore we consider also a degraded version of RVDD-basic in which we demosaic the raw frame before warping:

$$\widehat{u}_t = \mathcal{F}(\mathcal{W}_{t-1,t}\mathcal{D}(\widehat{u}^{\mathrm{M}}_{t-1}), \mathcal{D}(f_t)). \tag{6}$$

We refer to the former method as RVDD-basic-$\mathcal{DW}$ and to the latter as RVDD-basic-$\mathcal{WD}$. Propagating the raw and demosaicing before warping causes a drop of 0.15dB. Although this is not a negligible drop, it is rather small. This can be exploited in use cases in which there are limitations on the amount of information passed from one frame to the next. As expected, applying the warping on the raw domain causes a larger drop of around 0.25dB.

In total, propagating and warping raw frames accounts for 0.4dB out of the 2.2dB gap between the baseline JDD RVDD-basic and raw denoising RVD-basic followed by a demosaicing network. Thus most of the difference comes from working on the RGB domain and end-to-end training.

Interestingly, the improvement in performance does not only come from the 2/3 of the pixel values that are interpolated by the demosaicing. Table 1 also shows the raw PSNR, obtained by comparing the mosaiced RGB output $M\widehat{u}_t$ with the clean raw u^{M}_t. The performance is significantly higher for the RVDD-basic JDD network, which shows that working at the RGB resolution and training for RGB reconstruction benefits also the raw denoising task.

In Figure 3, we show the comparison between our JDD methods and the raw denoiser followed by a demosaicing network. The JDD results has better recovery of details and less color demosaicing artifacts.

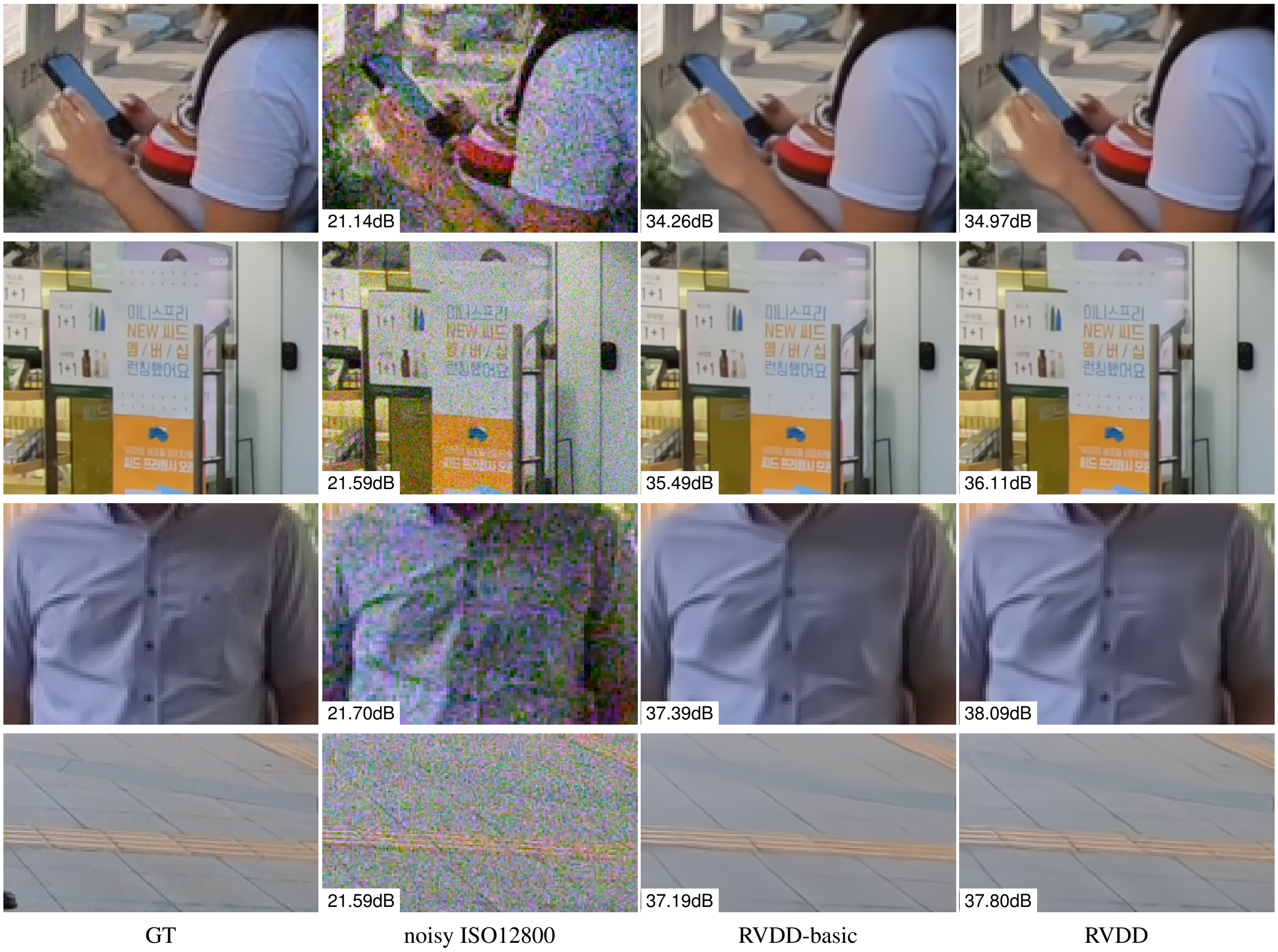

Figure 4: Results obtained with our method. We present two frameworks for handling the temporal information: recurrence only on the previous frame (RVDD-basic), or recurrence on the previous frame and features together with the use of the future frame (RVDD).

Ablation study. In Table 2, we show the effect of the different inputs to our RVDD network on our dataset with the two ISO levels. Adding the feature representation φ_{t-1}^{L} contributes 0.25dB and 0.3dB respectively for the low and high ISO. This makes intuitive sense: the feature map has C channels that can be used to give a richer representation of the spatial neighborhood of each pixel. The largest improvement comes however from adding the future raw frame f_{t+1}: compared to the baseline RVDD-basic, it gives a gain of 0.3dB for the ISO 3200 and 0.4dB for the ISO 12800 (in the linear RGB domain). The best results are obtained when we add both the feature recurrence and the future frame. The final gain compared with the baseline is then 0.47dB for the small ISO and 0.62dB for the highest one. In Figure 4 we compare the results obtained with the baseline (only frame recurrence) and with the best configuration (frame and feature recurrence and the use of future frame). We can see that the full RVDD is able to recover more details.

Comparison with others methods. In Table 3, we compare our method with the FastDVDnet JDD described in Section 3. One of the appealing characteristics of FastDVDnet is that it does not require motion estimation. However, the REDS dataset contains significant camera shake which is unfavorable to FastDVDnet. Thus we also consider a stabilized version of our dataset. This is a practical use case, as most mobile cameras are capable of performing some sort of motion stabilization. This will allow us to evaluate the impact of motion stabilization of the performance of different methods. In addition, we can test generalization across datasets with different motion statistics.

Since FastDVDnet is not a recurrent network, we include four non-recurrent versions of our network in the comparison: with and without warping (denoted by $\mathcal{W}$ in Table 3), and with and without the future frame f_{t+1}. We call these non-recurrent variants VDD. Finally, we add to the comparison the RVDD-basic as a recurrent baseline.

The best results in PSNR and SSIM are obtained by the networks with motion compensation, for both stabilized and non-stabilized datasets. The recurrent RVDD achieves the best performance in all cases, except when generalizing from the non-stabilized dataset to the stabilized. It is noteworthy that RVDD-basic, with only two input frames (the current frame f_t and the motion compensated previous output frame $\mathcal{W}_{t-1,t}\widehat{u}_{t-1}$), achieves a better performance than the non-recurrent VDD network with three motion compensated input frames (around 0.2dB). This shows the impact of frame recurrence in aggregating temporal information. When compared with the VDD without the future frame, the difference climbs to 0.7dB.

The networks without motion compensation are consistently worse in both datasets, although as expected, the performance gap is larger on the non-stabilized dataset. The gap between the best non motion compensated network and the worst with motion compensation is 1dB on the non-stabilized *vs.* 0.3dB on the stabilized.

For the VDD network, motion compensation allows to make better use of the additional temporal information when adding the future frame f_{t+1} to the inputs. With motion compensation, the PSNR gain is between 0.5dB and 0.7dB in all cases. Without motion compensation, there is still a small gain of around 0.2dB on the stabilized dataset, but there is no gain on the non-stabilized dataset and in fact, there might be a loss of around 0.2dB.

Finally, we can also evaluate the generalization ability of a network across changes in the motion statistics. To that aim, we compare the performance attained on a dataset A by a network trained on dataset A versus the same network trained on dataset B. With motion compensation this generalization gap is between 0.05dB and 0.07dB, regardless of the direction of the generalization (from stabilized to non-stabilized or viceversa). The exception is the full RVDD, which has worse generalization gap from the non-stabilized to the stabilized dataset (0.24dB and 0.13dB depending on the ISO). For the networks without motion compensation the generalization gap is larger. The largest one is for FastDVDnet-JDD on the non-stabilized dataset: 0.58dB. This is intuitive: when compensating for motion we are factoring out the motion in the dataset.

Improved architecture. We tested an modified U-Net taking into account the latest improvements in convolutional architecture design. We call the resulting architecture a ConvNeXt U-Net. It has the same structure as the original U-Net [49] with four main differences: (1) The 3×3 convolutions followed by ReLUs are replaced by ConvNeXt blocks [38] (see supplementary material for details). (2) A ConvNeXt block is inserted right after every downsampling and upsampling operation. (3) Three downsampling/upsampling operations are used instead of four. (4) At the end of the network, two additional ConvNeXt blocks are added at the finest scale.

This new architecture does not increase the number of FLOPS and has been proven to be very expressive for classification [38]. In addition, LayerScale [58] is used with a starting value of 0.1. Surprisingly, while we found that Batch Normalization [26] harmed the performance of the U-Net on our task, we found that LayerNorm did have a positive impact. Regarding LayerScale, we noticed that a too small initial value resulting in longer convergence time.

We compare both U-Nets on the baseline RVDD-basic and with the full network RVDD. For the baseline, both architectures reach the same performance. However, the training converged much faster with the ConvNeXt U-Net (about 30 epochs versus 100 epoch for the first architecture). A plot comparing the PSNR per epoch in our validation dataset for both architectures is available in the supplementary materials. For the full RVDD network, the ConvNeXt U-Net yields a gain of 0.2dB for the ISO 3200 and 0.36dB for the ISO 12800. Table 4 summarizes these results.

Real raw videos. In Figure 1 we show results obtained on real raw videos from the CRVD dataset for ISO 12800. The proposed RVDD recovers more details and is sharper. More results can be found in the supplementary material.

7. Conclusions

In this work we apply neural networks to the problem of video joint denoising and demosaicing for the first time. While related to image and burst JDD, the case of video has significant differences and enough relevance so as to deserve a separate treatment. In particular, recurrent neural networks such as the ones explored in our work are better suited for video than for bursts. We proposed a basic baseline network: a U-net where different inputs are concatenated, and we evaluated different configurations: inputting different number of frames, frame recurrent, feature recurrent and non-recurrent, motion compensation or not. In addition, we explore an adaptation to JDD of a state-of-the-art video denoising network, FastDVDnet, and compare its performance with those attained by the baseline U-net. The best results were obtained by the recurrent U-Net, yielding a strong baseline for video joint denoising and demosaicing. The main limitation of the proposed approach is its dependence on the optical flow. Ongoing work focuses on improving this aspect.

Acknowledgments Work partly financed by Office of Naval research grant N00014-17-1-2552 and MENRT. This work was performed using HPC resources from GENCI–IDRIS (grant 2022-AD011011801R2) and from the "Mésocentre" computing center of CentraleSupélec and ENS Paris-Saclay supported by CNRS and Région Île-de-France (http://mesocentre.centralesupelec.fr/).

References

[1] Pablo Arias and Jean-Michel Morel. Kalman filtering of patches for frame-recursive video denoising. In *The IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2019.

[2] Goutam Bhat, Martin Danelljan, Radu Timofte, Yizhen Cao, Yuntian Cao, Meiya Chen, Xihao Chen, Shen Cheng, Akshay Dudhane, Haoqiang Fan, et al. Ntire 2022 burst super-resolution challenge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1041–1061, 2022.

[3] Goutam Bhat, Martin Danelljan, Fisher Yu, Luc Van Gool, and Radu Timofte. Deep reparametrization of multi-frame super-resolution and denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2460–2470, 2021.

[4] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T Barron. Unprocessing images for learned raw denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11036–11045, 2019.

[5] Antoni Buades and Joan Duran. Cfa video denoising and demosaicking chain via spatio-temporal patch-based filtering. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(11):4143–4157, 2019.

[6] Priyam Chatterjee, Neel Joshi, Sing Bing Kang, and Yasuyuki Matsushita. Noise suppression in low-light images through joint denoising and demosaicing. In *CVPR 2011*, pages 321–328. IEEE, 2011.

[7] Huaian Chen, Yi Jin, Kai Xu, Yuxuan Chen, and Changan Zhu. Multiframe-to-multiframe network for video denoising. *IEEE Transactions on Multimedia*, 24:2164–2178, 2022.

[8] Miguel Colom and Antoni Buades. Analysis and Extension of the Ponomarenko et al. Method, Estimating a Noise Curve from a Single Image. *Image Processing On Line*, 3:173–197, 2013.

[9] Valéry Dewil, Jérémy Anger, Axel Davy, Thibaud Ehret, Gabriele Facciolo, and Pablo Arias. Self-supervised training for blind multi-frame video denoising. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2724–2734, 2021.

[10] Weishong Dong, Ming Yuan, Xin Li, and Guangming Shi. Joint demosaicing and denoising with perceptual optimization on a generative adversarial network. *arXiv preprint arXiv:1802.04723*, 2018.

[11] Thibaud Ehret, Axel Davy, Pablo Arias, and Gabriele Facciolo. Joint demosaicking and denoising by fine-tuning of bursts of raw images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8868–8877, 2019.

[12] Thibaud Ehret and Gabriele Facciolo. A study of two cnn demosaicking algorithms. *Image Processing On Line*, 9:220–230, 2019.

[13] Thibaud Ehret, Jean-Michel Morel, and Pablo Arias. Non-local kalman: A recursive video denoising algorithm. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3204–3208. IEEE, 2018.

[14] Sina Farsiu, Michael Elad, and Peyman Milanfar. Multiframe demosaicing and super-resolution of color images. *IEEE transactions on image processing*, 15(1):141–159, 2005.

[15] Alessandro Foi, Mejdi Trimeche, Vladimir Katkovnik, and Karen Egiazarian. Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. *IEEE Transactions on Image Processing*, 17(10):1737–1754, 2008.

[16] Dario Fuoli, Shuhang Gu, and Radu Timofte. Efficient video super-resolution through recurrent latent space propagation. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3476–3485. IEEE, 2019.

[17] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

[18] Bart Goossens, Hiep Luong, Jan Aelterman, Aleksandra Pizurica, and Wilfried Philips. An overview of state-of-the-art denoising and demosaicking techniques: toward a unified framework for handling artifacts during image reconstruction. In *Image Sensor Workshop*, 2015.

[19] Shi Guo, Zhetong Liang, and Lei Zhang. Joint denoising and demosaicking with green channel prior for real-world burst images. *IEEE Transactions on Image Processing*, 30:6930–6942, 2021.

[20] Shi Guo, Zhetong Liang, and Lei Zhang. Joint denoising and demosaicking with green channel prior for real-world burst images. *IEEE Transactions on Image Processing*, 30:6930–6942, 2021.

[21] Shi Guo, Xi Yang, Jianqi Ma, Gaofeng Ren, and Lei Zhang. A differentiable two-stage alignment scheme for burst image reconstruction with large shift. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17472–17481, 2022.

[22] Felix Heide, Markus Steinberger, Yun-Ta Tsai, Mushfiqur Rouf, Dawid Pająk, Dikpal Reddy, Orazio Gallo, Jing Liu, Wolfgang Heidrich, Karen Egiazarian, et al. Flexisp: A flexible camera image processing framework. *ACM Transactions on Graphics (TOG)*, 33(6):231, 2014.

[23] Keigo Hirakawa and Thomas W Parks. Joint demosaicing and denoising. *IEEE Transactions on Image Processing*, 15(8):2146–2157, 2006.

[24] Cong Huang, Jiahao Li, Bin Li, Dong Liu, and Yan Lu. Neural compression-based feature learning for video restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5872–5881, June 2022.

[25] Tao Huang, Fang Fang Wu, Weisheng Dong, Guangming Shi, and Xin Li. Lightweight deep residue learning for joint color image demosaicking and denoising. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 127–132. IEEE, 2018.

[26] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[27] Takashi Isobe, Xu Jia, Shuhang Gu, Songjiang Li, Shengjin Wang, and Qi Tian. Video super-resolution with recurrent structure-detail network. In *European conference on computer vision*, pages 645–660. Springer, 2020.

[28] Qiyu Jin, Gabriele Facciolo, and Jean-Michel Morel. A review of an old dilemma: Demosaicking first, or denoising first? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 514–515, 2020.

[29] Qiyu Jin, Yu Guo, Jean-Michel Morel, and Gabriele Facciolo. A mathematical analysis and implementation of residual interpolation demosaicking algorithms. *Image Processing On Line*, 11:234–283, 2021.

[30] James E. Adams Jr. and John F. Hamilton Jr. Adaptive color plane interpolation in single sensor color electronic camera, US Patent 5,629,734, Nov. 1996.

[31] Ossi Kalevo and Henry Rantanen. Noise reduction techniques for bayer-matrix images. In *Sensors and Camera Systems for Scientific, Industrial, and Digital Photography Applications III*, volume 4669, pages 348–359. SPIE, 2002.

[32] Daniel Khashabi, Sebastian Nowozin, Jeremy Jancsary, and Andrew W Fitzgibbon. Joint demosaicing and denoising via learned nonparametric random fields. *IEEE Transactions on Image Processing*, 23(12):4968–4981, 2014.

[33] Tae Hyun Kim, Mehdi S. M. Sajjadi, Michael Hirsch, and Bernhard Scholkopf. Spatio-temporal transformer network for video restoration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[34] Teresa Klatzer, Kerstin Hammernik, Patrick Knobelreiter, and Thomas Pock. Learning joint demosaicing and denoising based on sequential energy minimization. In *2016 IEEE International Conference on Computational Photography (ICCP)*, pages 1–11. IEEE, 2016.

[35] Filippos Kokkinos and Stamatios Lefkimmiatis. Iterative joint image demosaicking and denoising using a residual denoising network. *IEEE Transactions on Image Processing*, 28(8):4177–4188, 2019.

[36] Bruno Lecouat, Jean Ponce, and Julien Mairal. Lucas-kanade reloaded: End-to-end super-resolution from raw image bursts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2370–2379, 2021.

[37] Lin Liu, Xu Jia, Jianzhuang Liu, and Qi Tian. Joint demosaicing and denoising with self guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2240–2249, 2020.

[38] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022.

[39] Rastislav Lukac and Konstantinos N Plataniotis. Adaptive spatiotemporal video demosaicking using bidirectional multistage spectral filters. *IEEE Transactions on Consumer Electronics*, 52(2):651–654, 2006.

[40] Matteo Maggioni, Yibin Huang, Cheng Li, Shuai Xiao, Zhongqian Fu, and Fenglong Song. Efficient multi-stage video denoising with recurrent spatio-temporal fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3466–3475, 2021.

[41] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P Srinivasan, and Jonathan T Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16190–16199, 2022.

[42] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In

The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, June 2019.

[43] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *CVPR Workshops*, June 2019.

[44] Piotr Kopa Ostrowski, Efklidis Katsaros, Daniel Węsierski, and Anna Jezierska. Bp-evd: Forward block-output propagation for efficient video denoising. *IEEE Transactions on Image Processing*, 31:3809–3824, 2022.

[45] Dmitriy Paliy, Mejdi Trimeche, Vladimir Katkovnik, and Sakari Alenius. Demosaicing of noisy data: spatially adaptive approach. In *Image Processing: Algorithms and Systems V*, volume 6497, pages 179–190. SPIE, 2007.

[46] Sung Hee Park, Hyung Suk Kim, Steven Lansel, Manu Parmar, and Brian A Wandell. A case for denoising before demosaicking color filter array data. In *2009 Conference Record of the Forty-Third Asilomar Conference on Signals, Systems and Computers*, pages 860–864. IEEE, 2009.

[47] Naama Pearl, Tali Treibitz, and Simon Korman. Nan: Noise-aware nerfs for burst-denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12672–12681, June 2022.

[48] Nikolay Ponomarenko, Vladimir V. Lukin, Mikhail Zriakhov, Arto Kaarna, and Jaakko T. Astola. An automatic approach to lossy compression of AVIRIS images. In *Geoscience and Remote Sensing Symposium, 2007. IGARSS 2007. IEEE International*, pages 472–475. IEEE, 2007.

[49] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[50] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018.

[51] Javier Sánchez. Comparison of motion smoothing strategies for video stabilization using parametric models. *Image Processing On Line*, 7:309–346, 2017.

[52] Javier Sánchez and Jean-Michel Morel. Motion smoothing strategies for 2d video stabilization. *SIAM Journal on Imaging Sciences*, 11(1):219–251, 2018.

[53] Dev Yashpal Sheth, Sreyas Mohan, Joshua Vincent, Ramon Manzorro, Peter A. Crozier, Mitesh M. Khapra, Eero P. Simoncelli, and Carlos Fernandez-Granda. Unsupervised deep video denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021.

[54] Lu Sun, Weisheng Dong, Xin Li, Jinjian Wu, Leida Li, and Guangming Shi. Deep maximum a posterior estimator for video denoising. *International Journal of Computer Vision*, 129(10):2827–2845, 2021.

[55] Nai-Sheng Syu, Yu-Sheng Chen, and Yung-Yu Chuang. Learning deep convolutional networks for demosaicing. *arXiv preprint arXiv:1802.03769*, 2018.

[56] Runjie Tan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. Color image demosaicking via deep residual learning. In *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, pages 793–798, 2017.

[57] Matias Tassano, Julie Delon, and Thomas Veit. Fastdvdnet: Towards real-time deep video denoising without flow estimation. In *The IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1354–1363, June 2020.

[58] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 32–42, 2021.

[59] Gregory Vaksman, Michael Elad, and Peyman Milanfar. Patch craft: Video denoising by deep modeling and patch matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2157–2166, October 2021.

[60] Bartlomiej Wronski, Ignacio Garcia-Dorado, Manfred Ernst, Damien Kelly, Michael Krainin, Chia-Kai Liang, Marc Levoy, and Peyman Milanfar. Handheld multi-frame super-resolution. *ACM Transactions on Graphics (TOG)*, 38(4):1–18, 2019.

[61] Xiaolin Wu and Lei Zhang. Temporal color video demosaicking via motion estimation and data fusion. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(2):231–240, 2006.

[62] Wenzhu Xing and Karen Egiazarian. End-to-end learning for joint image demosaicing, denoising and super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3507–3516, 2021.

[63] Xiangyu Xu, Muchen Li, Wenxiu Sun, and Ming-Hsuan Yang. Learning spatial and spatio-temporal pixel aggregations for image and video denoising. *IEEE Transactions on Image Processing*, 29:7153–7165, 2020.

[64] Huanjing Yue, Cong Cao, Lei Liao, Ronghe Chu, and Jingyu Yang. Supervised raw video denoising with a benchmark dataset on dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2301–2310, 2020.

[65] Christopher Zach, Thomas Pock, and Horst Bischof. A duality based approach for realtime tv-l 1 optical flow. In *Joint pattern recognition symposium*, pages 214–223. Springer, 2007.

[66] Lei Zhang, Weisheng Dong, Xiaolin Wu, and Guangming Shi. Spatial-temporal color video reconstruction from noisy cfa sequence. *IEEE transactions on circuits and systems for video technology*, 20(6):838–847, 2010.

[67] Lei Zhang, Rastislav Lukac, Xiaolin Wu, and David Zhang. Pca-based spatially adaptive denoising of cfa images for single-sensor digital cameras. *IEEE transactions on image processing*, 18(4):797–812, 2009.