Supplement to Dancing under the stars: video denoising in starlight

Kristina Monakhova
UC Berkeley

Stephan R. Richter
Intel Labs

Laura Waller
UC Berkeley

Vladlen Koltun
Intel Labs

Abstract

In this document we include supplementary materials to Dancing under the stars: video denoising in starlight. We provide additional implementation details and visual results on still images from our dataset.

1. Additional Implementation Details

1.1. Noise Generator and Discriminator

First, we provide additional details about the noise generator and discriminator. In our noise generator, we include a periodic noise component. This periodic noise component is modeled as follows in the frequency domain as:

$$n[M,N] = \mathcal{F}^{-1}\begin{cases} X_1, & \text{if } N = 0 \\ X_2 + X_3 j, & N = N_t/4 \\ X_2 - X_3 j, & N = 3N_t/4 \\ 0, & \text{otherwise} \end{cases}$$

Where X_1, X_2, and X_3 are zero mean Gaussian random variables with optimized variances λ_{f1}, λ_{f2}, λ_{f3}, N_t is the total number of columns in the image, and M, N index the rows/columns of the image. This essentially corresponds to adding a 1 or 2 pixel period sinusoidal pattern to the image with a random amplitude that is determined by the optimized variance parameter. We demonstrate the effect of this noise in Figure 1, showing a central slice of the Fourier transform of the clean vs. noisy images. We can see that our full model better matches the real noise than our partial model which does not consider the periodic noise component.

For the CNN in our noise generator, we use a standard 2D residual U-Net architecture, with 4 input and output channels, 4 upsampling and downsampling layers, stride-2 convolutional downsampling layers, stride-2 transpose convolutional upsampling layers, and SeLU activations. The number of channels in our 4 downsampling and upsampling layers are 32, 64, 128, and 256.

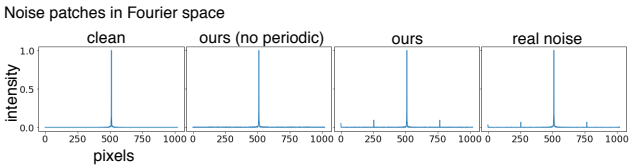

Figure 1. Fourier transform of clean and noisy patches, showing the prominent spike in Fourier space that we see in the real noisy images. Our full model captures this behaviour.

We initialize our shot, read, row, and row temporal noise parameters to 2e-1, 2e-2, and 2e-3, and 2e-3 respectively. We initialize our uniform noise parameter to 1e-1, and our periodic parameters to 5. In general, we note that the initialized value of these parameters did not seem to effect the final converged value as long as the initial values were small.

Our discriminator's architecture is outlined in Fig. 2. We feed in images with a patch size of 64 into the discriminator during training. We use an Adam optimizer [1] with a learning rate of 0.0002, with $\beta_1 = 0.5$ and $b_2 = 0.999$ for both the generator and discriminator. For each experiment in our generator ablation study, we feed both the noisy patch as well as the Fourier transform of the noisy patch into the discriminator, which we found resulted in better performance than using either the image or the Fourier transform of the image alone. For the final two comparisons in the ablation (all the noise parameters with U-Net and all the noise parameters without the U-Net) we use only the Fourier transform of the image in the discriminator, which resulted in the best performance given those parameters. In all experiments, we add an LPIPS loss to our generator loss. We take a gradient step on the generator after every 5 gradient steps on the discriminator.

1.2. Denoiser details

We base our denoiser on FastDVDnet [3]. We modify the FastDVDnet architecture in two ways. First, we increase the number of channels to 4 instead of 3 to facilitate processing our RAW images. Next, we modify the denoiser blocks.

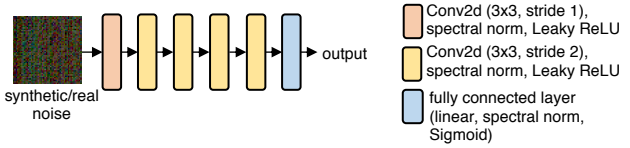

Figure 2. Discriminator architecture. We use our discriminator during our noise generator training.

Stage	N_{br}	N_c	N_m	N_{bl}
1	1	[64]	1	[4]
2	2	[18, 36]	1	[4,4]
3	3	[18, 36, 72]	3	[4,4,4]
4	4	[18, 36, 72, 144]	3	[4,4,4,4]

Table 1. HRNet architecture

The original implementation of FastDVDnet uses a U-Net architecture for the denoising blocks. We replace this architecture with HRNet blocks. In our raw high gain, low light videos, we often see flashing and differences in colors between frames (Figure 3). Experimentally, we found that using HRNet blocks reduces the flickering across frames that we see at our lowest light settings. Figure 3 shows an example of this with a FastDVDNet denoiser using U-Net blocks vs. as FastDVDnet denoising with HRNet blocks. When plotting the mean intensity over time, we can see that version with HRNet blocks has less variance in the intensity, effectively smoothing out the flickering over frames, whereas the FastDVDNet with U-Net blocks is not effective at reducing flickering, having a higher variance in the mean intensity over time.

Following from FastDVDnet, our denoiser architecture consists of two denoising blocks. Each block takes in 3 images with 4 channels each (12 channels total) and outputs a single image with 4 channels. We use an HRNet designed for semantic segmentation [2,4,5] and slightly modify it to work on our images by replacing the initial stride-2 convolutions to stride-1 convolutions. Our HRNet has 4 stages. The first stage consists of a Bottleneck block, while the remaining stages consists of Basic blocks. We summarize the number of branches (N_{br}), number of channels (N_c), number of modules (N_m), and number of blocks (N_{bl}) in each stage in the Table 1.

1.3. Camera details

For all noisy sequences, we use the highest camera gain: 16× column amplifier gain), 6dB CDS Gain, and 1023 VGA gain. In addition, all images are stored as RAW unprocessed images with RGB+NIR channels. The clean images were taken at 1× column amplifier gain, 6dB CDS Gain, and 10 VGA gain. For all images, the exposure on the clean/noisy images was set to approximately match their intensities

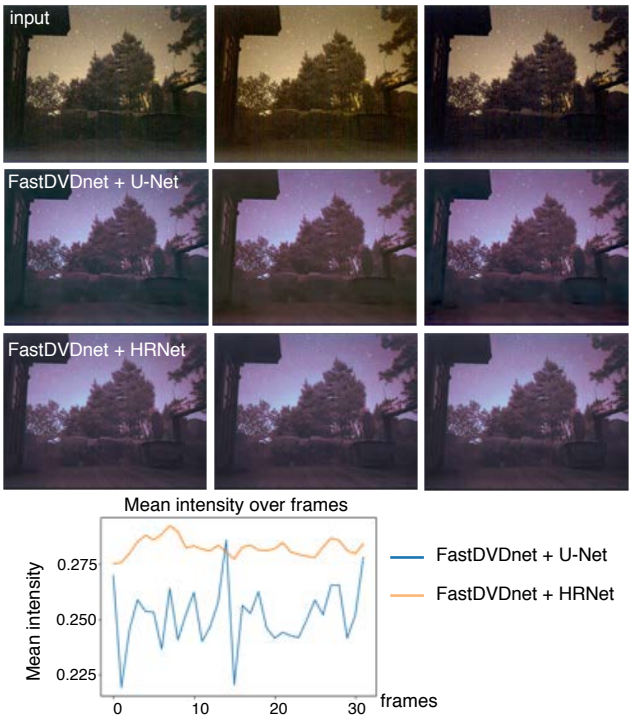

Figure 3. Architecture comparison: FastDVDnet + Unet vs FastDVDnet + HRnet. Here we can see that original FastDVDnet results in more flickering between frames than our modified FastDVDnet (with HRnet).

(1000×). For all sets of images, a scalar offset to correlate the mean intensity for the clean/noisy pairs was calculated and applied to each clean image to match the mean to that of the noisy images.

2. Additional Results

2.1. Simplified Noise Model results

Next, we perform an additional ablation in which we keep our denoiser constant and use a different noise generator to synthesize our noisy video clips. We compare using our full noise generator against our full noise generator without the U-Net and with only read, shot, and uniform noise in Table 2, showing the performance on our stills dataset. We can see that our full model with the U-Net performs the best. We anticipate that the U-Net is able to learn additional features of the noise that we do not explicitly model (e.g. chromatic effects in the noise) and perhaps augment any simplifications in our noise mode (e.g. using a heteroscedastic Gaussian noise model rather than a Poisson model for shot noise). Given better synthetic noise, our denoiser can successfully tackle sensor-specific noise and produce the best denoised images.

Method	PSNR	SSIM	LPIPS
Ours (Shot+read+uniform)	23.8	0.861	0.111
Ours no U-Net	25.5	0.910	0.115
ours (full)	**27.7**	**0.931**	**0.078**

Table 2. Performance on still images from test set.

2.2. Stills Results

Next we provide additional images to showcase our results on our stills dataset. We compare against V-BM4D, L2SID, N2S, and FastDVDNet. We compare both against pretrained L2SID as well as L2SID retrained using our stills dataset. Two example images are shown in Figure 4. Here we can see that V-BM4D and N2S both have significant line artifacts throughout the images. Pretrained L2SID has issues with color, since our camera has an additional NIR channel rather than only RGB channels, and also blurs out features due to differences in our camera noise. When we pretrain L2SID using our own data, the performance improves substantially for still images as expected. Since L2SID is a single-image method and ours uses 5 images to collaboratively denoise an image, we still outperform L2SID in dark regions of the image (e.g. the text on the cans in clip 2). Furthermore, retrained L2SID results in severe flickering and poor performance on moving videos (see attached video clips). Similarly, pretrained FastDVDNet contains stripe artifacts and has reduced resolution since it is trained using a Gaussian noise model. Our method closely matches the ground truth images, maintaining the image features while suppressing the noise.

See attached supplemental video for a video comparison between our method, V-BM4D, L2SID (retrained on our stills data), and FastDVDNet. All videos are downsized by 2× from the full resolution and cropped by 880×630 pixels (full resolution is 2160×1280). In addition, we provide a video with a compilation of denoised clips from our submillilux dataset. In these videos, we demonstrate the performance of our denoiser at the most challenging low light setting with significant motion.

2.3. Perceptual Experiments

We perform a perceptual experiment with blind randomized A/B tests between our method, V-BM4D, FastDVDNet, and L2SID. We show 10 clips from our submillilux dataset. Each clip is 30 frames long and is cropped to a 400×400 region which shows significant motion. During the experiment, we show 2 video clips side by side in a randomized order and workers are asked which clip they prefer. We run 300 comparisons in total with 10 workers. The results are summarized below:

- 95.0 [+/- 4.27]% prefer our method over FastDVDNet.
- 99.0 [+/- 1.95]% prefer our method over L2SID.
- 97.0 [+/- 3.34]% prefer our method over V-BM4D.

As we can see, in all the experiments, video clips produced by our method are preferred over alternative methods by a large margin.

References

[1] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 1

[2] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019. 2

[3] Matias Tassano, Julie Delon, and Thomas Veit. Fastdvdnet: Towards real-time deep video denoising without flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1354–1363, 2020. 1

[4] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *TPAMI*, 2019. 2

[5] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. 2020. 2

Figure 4. Denoising comparison on two noisy bursts of still objects from our test set.