Efficient Multi-Stage Video Denoising with Recurrent Spatio-Temporal Fusion
Supplementary Materials

Matteo Maggioni*, Yibin Huang*, Cheng Li*, Shuai Xiao, Zhongqian Fu, Fenglong Song
Huawei Noah's Ark Lab
{matteo.maggioni, huangyibin1, licheng89, xiaoshuai7, fuzhongqian, songfenglong}@huawei.com

1. Implementation Aspects

1.1. Learnable Invertible Transforms

Color Transform. The $C \times C$ color transform matrix is analogous to a YUV transformation for RGB domain. A YUV transform matrix has size $C = 3$, however the proposed model is designed for raw data, thus in our case the matrix will have size $C = 4$, in order to transform each color in the CFA Bayer pattern (e.g., RG_1G_2B). Practically the matrix is defined as [2]

$$M = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ -0.5 & 0.5 & 0.5 & -0.5 \\ 0.65 & 0.2784 & -0.2784 & -0.65 \\ -0.2784 & 0.65 & -0.65 & 0.2784 \end{bmatrix} = \begin{bmatrix} Y \\ U \\ V \\ W \end{bmatrix} \quad (1)$$

where each row has unit norm and corresponds to a different color transform basis. The luminance component Y can be easily recognized in the first row of (1), and unsurprisingly it corresponds to an (energy-preserving) average of the four input color channels. In our context, the matrix M will be used to initialize the $1 \times 1 \times C \times C$ kernel of a (point-wise) convolutional layer.

Frequency Transform. As initialization value for our learnable frequency transform we use filters obtained by standard wavelet families. In fact, each wavelet type has a pair of decomposition filters, a low-pass ψ_L and a high-pass ψ_H, as well as a complementary pair of reconstruction filters, again, a low-pass ϕ_L and a high-pass ϕ_H. These are all real 1-D filters of size $1 \times n$, being $n \in \mathbb{N}^+$ an even integer value. We use these filters to generate the corresponding $n \times n$ convolutional kernels. For example, the 2-D LL decomposition kernel is obtained as $\psi_L \otimes \psi_L$ being $\otimes$ the outer product. We show all components involved in the learning and application of the frequency transform in Fig. 1.

1.2. Models

VBM4D. VBM4D [5] is a traditional algorithm originally designed to remove independent and identically distributed zero-mean Gaussian noise in grayscale or RGB video. However, in our experiments, we apply VBM4D on sRGB videos generated by an ISP [8] applied to the noisy raw data. Thus the noise will be not independent, not identically distributed, and not white. These are not ideal conditions for VBM4D, but we optimize its σ parameter, which can be used to control the amount of denoising, to maximize the PSNR of the validation data. We simply perform a grid search to find the best σ for each ISO and each dataset.

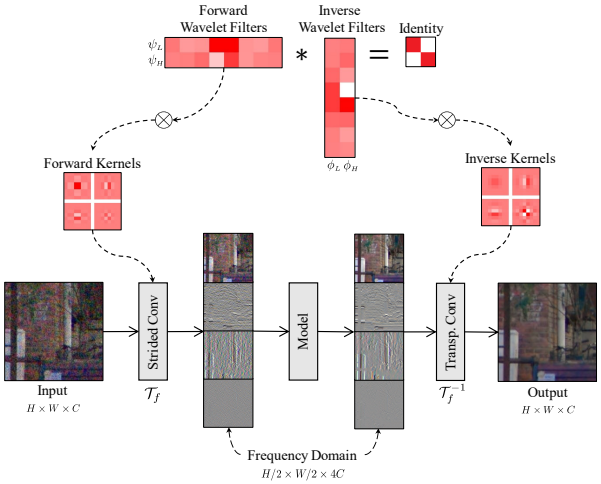


Figure 1: Frequency transform: convolutional kernels corresponds to the outer product $\otimes$ of the learned filters.

FastDVDnet. We use the original FastDVDnet implementation provided by the authors [6]. FastDVDnet is designed for Gaussian noise removal and uses a uniform noise map corresponding to the variance of the distribution as additional input of the network. Since we deal with signal-dependent noise, we replace the uniform map with the variance map computed according to the raw noise model defined in (2) of the main paper. In order to decrease model complexity, we reduce the number of channels. Specifically, in the 82.61 GFLOPs version, we use 8 channels in the input layers, 16 channels in the highest-resolution scale, and 24 everywhere else. In the 22.16 GFLOPs version we use 8 channels everywhere.

	Operation	Kernel Size	Filters	Output Size	Comment
	Input	-	-	$H \times W \times 4$	Current noisy frame.
	Conv2D	$1 \times 1 \times 4$	4	$H \times W \times 4$	Color transform.
	StridedConv2D	$2 \times 2 \times 4$	16	$H/2 \times W/2 \times 16$	Frequency transform (stride 2).
Fusion	Sub + Abs	-	-	$H/2 \times W/2 \times 4$	Absolute difference of low-pass subbands of current and previous fused frames (4 channels).
Fusion	Concat	-	-	$H/2 \times W/2 \times 6$	Concat low-pass absolute difference (4 channels), nearest neighbor upsampling of fusion weights from lower scale (1 channel), and variance map of noisy frame (1 channel)
Fusion	Conv2D + ReLU	$3 \times 3 \times 6$	16	$H/2 \times W/2 \times 16$	Input layer of fusion network.
Fusion	Conv2D + ReLU	$3 \times 3 \times 16$	16	$H/2 \times W/2 \times 16$	Hidden layer (can be repeated).
Fusion	Conv2D	$3 \times 3 \times 16$	1	$H/2 \times W/2 \times 1$	Fusion output.
Fusion	Sigmoid	-	-	$H/2 \times W/2 \times 1$	Fusion weights.
Fusion	Mul + Add	-	-	$H/2 \times W/2 \times 16$	Fusion of current noisy (16 channels) and previous fused (16 channels) using fusion weights (1 channel).
Denoising	Concat	-	-	$H/2 \times W/2 \times 25$	Concat fused output (16 channels), low-pass subband of current frame (4 channels), denoising output at lower scale after inverse frequency transform (4 channels), variance map of fused frame (1 channel)
Denoising	Conv2D + ReLU	$3 \times 3 \times 25$	16	$H/2 \times W/2 \times 16$	Input layer of denoising network.
Denoising	Conv2D + ReLU	$3 \times 3 \times 16$	16	$H/2 \times W/2 \times 16$	Hidden layer (can be repeated).
Denoising	Conv2D	$3 \times 3 \times 16$	16	$H/2 \times W/2 \times 16$	Denoising output.
Refinement	Concat	-	-	$H/2 \times W/2 \times 33$	Concat fusion output (16 channels), denoising output (16 channels), and variance map (1 channel).
Refinement	Conv2D + ReLU	$3 \times 3 \times 33$	16	$H/2 \times W/2 \times 16$	Input layer of refinement network.
Refinement	Conv2D + ReLU	$3 \times 3 \times 16$	16	$H/2 \times W/2 \times 16$	Hidden layer (can be repeated).
Refinement	Conv2D	$3 \times 3 \times 16$	16	$H/2 \times W/2 \times 16$	Refinement output.
Refinement	Sigmoid	-	-	$H/2 \times W/2 \times 1$	Refinement weights.
Refinement	Mul + Add	-	-	$H/2 \times W/2 \times 16$	Refinement of denoising output (16 channels) using current fused (16 channels) and refinement weights (1 channel).
	TransConv2D	$2 \times 2 \times 16$	4	$H \times W \times 4$	Inverse frequency transform (2× upsampling).
	Conv2D	$1 \times 1 \times 4$	4	$H \times W \times 4$	Inverse color transform.
	Output	-	-	$H \times W \times 4$	Final output frame.

Table 1: Architecture of the proposed EMVD method.

EDVR. We build EDVR baseline as in [7], with five residual blocks in PCD alignment module and 40 residual blocks in the reconstruction module. The number of filters is 128. Since we are dealing with raw denoising, we need to modify the input and output channels to be four, i.e. the number of colors in the CFA (for example RG_1G_2B). When constructing a smaller network we reduce the number of filters to 48. Also we remove the global residual connection because it has been shown that it makes convergence of low-capacity networks more difficult [4].

RViDeNet. In our experiments, we use the publicly available RViDeNet model[1]. We build a lower-complexity version of RViDeNet by reducing the capacity of all sub-networks in the original architecture. Specifically, in the pre-denoising module we reduce the number of scales to 2. In the other sub-networks, we use a single group of deformable convolutions, a single residual block in front of the alignment module, and two residual blocks following the temporal fusion. All convolutions have 16 channels. We train this model as suggested in the original paper [8], with only the raw reconstruction term in the loss.

EMVD. In Table 1 we describe the proposed EMVD architecture, listing all operations used to process a frame in the input video. For clarity we separate the three processing stages, as well as the initial and final transformations. A diagram is shown in Fig. 2 of the main paper.

[1] https://github.com/cao-cong/RViDeNet

Wavelet Type	Kernel Size	Δ PSNR
Haar	2×2	**0.0**
Symlets 2	4×4	-0.01
Daubechies 2	4×4	-0.19
Daubechies 3	6×6	-0.09
Biorthogonal 3.1	4×4	-0.15
Biorthogonal 1.3	6×6	-0.05
Rev. Biorthogonal 3.1	4×4	-0.25
Rev. Biorthogonal 1.3	6×6	-0.08

(a) Frequency transform initialization.

$\mathcal{T}_c$	$\mathcal{T}_f$	Learnable	Invertible	Δ PSNR
M	Haar	✓	✓	**0.0**
M	Haar	✓	×	-0.10
M	Haar	×	✓	-0.11
M	Random	✓	×	-2.77
Random	Haar	✓	✓	-0.45
Random	Haar	✓	×	-2.36
Random	Random	✓	✓	-0.77
Random	Random	✓	×	-3.45

(b) Transform settings.

Table 2: Ablation of proposed EMVD model using different settings for the learnable color and frequency transforms.

2. Additional Experiments

Learnable Transforms. In Table 2a, we analyze the effect of using different wavelet families as initialization of the frequency transform. The low-complexity EMVD model (5.38 GFLOPs) described in the main paper is highlighted in yellow in the tables, and it is used here as baseline for these experiments. As one can see, Haar kernels, despite having the smallest size, provide the best performance. However we do not observe significant differences when using some of the other wavelet types, even when the size of their kernels is larger. In Fig. 3 we visualize the frequency transform coefficients for all experiments. In each plot, we show both the learned transform coefficients as well as the corresponding wavelet coefficients used for initialization. We observe that the low-pass filters are relatively similar in all cases, but the differences in the high-pass filters are quite pronounced.

In Table 2b, we analyze different combinations for color and frequency transform initialization, and whether or not the transforms are set to be trainable or invertible. Note that since both transforms are implemented as convolutions, we can also analyze the result of using random initialization weights. We note that performance drops by up to 3.45dB PSNR when the color transform $\mathcal{T}_c$ is randomly initialized and invertibility loss is disabled. Enabling invertibility loss leads to an increase in performance, but PSNR is still 0.77dB lower than baseline if $\mathcal{T}_f$ is random and 0.45dB if $\mathcal{T}_f$ is initialized with Haar. Differently, when $\mathcal{T}_c$ is initialized with M (1) and $\mathcal{T}_f$ is initialized with Haar, both invertibility and learnable properties only provide 0.1dB increase in PSNR. Such relatively modest improvement can be explained by the filters shown in Fig. 4. As a matter of fact, whenever the transforms are initialized with M and Haar, the learned filters are remarkably similar even when invertibility is disabled. This demonstrates that the transforms learned by the network are always very close to be invertible. Therefore we can argue that invertibility is a desirable and optimal property of the transform operators.

Fusion. In Fig. 2, we show a set of frames from a single video sequence. From top to bottom we show the fused frames $\bar{y}_t$, the fusion weights γ_t (indicating dynamic regions in red), the noise variance $\hat{\sigma}_t^2$ in the noisy frame, and the variance $\bar{\sigma}_t^2$ of the noise in the fused frame recursively computed as defined in (9) from the main paper. Observe how the variance in regions where motion can be compensated is reducing over time. Also, it is interesting to note that the weights in the background will tend to zero (blue values) as the sequence progresses, thus indicating that more influence is given to the previous fused frame. This is intuitively correct, because ideally at frame t the weights γ_t given to the current frame should be proportional to $\frac{1}{t}$, whereas the weight $\bar{\gamma}_{t-1}$ given to the previous frame should be proportional to $1 - \frac{1}{t}$, therefore $\bar{\gamma}_{t-1} \gg \gamma_t$ as $t \to \infty$.

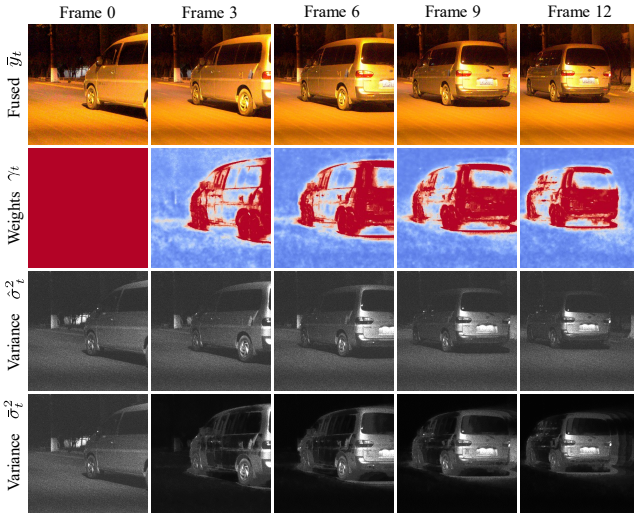


Figure 2: The variance of the fused frames is computed recursively with the fusion weights.

In Fig. 5a, we show some examples of fusion weights for different sequences. In the figure we show, from left to right, the previous fused image, the current noisy, the fusion weights corresponding to the current noisy image, and

finally the output of the fusion. The weights are color coded such that red corresponds to the most dynamic regions in the image. In those regions, fusion is effectively disabled as motion cannot be compensated, and the output fused image will be equal to the current noisy. Differently in blue we denote those regions where fusion has the highest effect. As one can see, the noise in the output image is dependent on the fusion weights, and it is larger within highly dynamic regions. We remark that even if the fusion network is not explicitly supervised, the output is still clearly interpretable.

Refinement. In Fig. 5b we visualize few more examples of frames obtained after each individual stage in the proposed EMVD. Particularly, from left to right, we show the noisy frame, fused frame, denoised frame, refinement weights applied to the fused image, and then the final (refined) output image. Again the refinement weights have a clear and interpretable behavior, as we can easily understand that the objective of the refinement stage is to identify and extract high-frequency information from the fused image in order to add them onto the denoised one. As a matter of fact, the final result is the optimal combination of these two images, as the noise is effectively suppressed without compromising the quality of edges, fine details, and textures in the (refined) output image.

Finally, in Fig. 5c, we show output images at different frames of a single test sequence in the CRVD dataset [8]. It is interesting to observe the progression of the proposed EMVD in each individual stage, as the image quality steadily increases as more frames are processed. In fact, the noise is progressively reduced in the fused image, which in turn positively affects the subsequent denoising and refinement tasks. This is also clearly visible from the fusion and refinement weights, which both get more accurate with time. In particular, we notice how the refinement weights are able to identify an increasing amount of high-frequency information as the signal-to-noise-ratio in the fused image improves. This figure visualized the interdependence of spatial (sequential) and temporal (recurrent) tasks in the proposed EMVD. However, despite this complex and non-linear behavior, we stress once again that the proposed model is straightforward to train as no intermediate supervision is required for its successful convergence.

Visual Comparisons. We show additional results on the CRVD [8] dataset. Objective evaluation is reported in Table 3 and Fig. 6 of the main paper. In Fig. 6 we show comparisons against reference state-of-the-art methods, namely VBM4D [5], FastDVDnet [6], EDVR [7], and the recently proposed RViDeNet [8]. As one can see, despite its significantly lower complexity, the proposed EMVD is able to achieve similar –if not better– denoising results among all compared methods. Different dynamic (yellow crops) and static (red crops) regions are highlighted for each test case. In Fig. 7 we focus on the low-complexity scenario. In this case, we compare our EMVD against the best low-complexity models, namely FastDVDnet [6]. The proposed method decidedly achieves the best performance, hence corroborating the superior objective performance reported in the main paper.

3. Noise Model Estimation

The standard Gaussian-Poissonian noise model for raw data [3] can be estimated using a number of calibration images captured at various ISO levels. For the sake of simplicity the ISO is assumed to be the only camera parameter that is affecting the noise distribution. With this, we can estimate the noise variance, formalized in (2) of the main paper, using noise estimation methods that apply robust estimators of scale (such as the Median of Absolute Deviations – MAD) on high-frequency transform-domain coefficients obtained by decorrelating transform operators (such as DCT or Wavelet). Ideally, estimation should be applied to patches whose underlying intensity is as uniform as possible in order to avoid interference caused by high-frequency response of edges or textures [1].

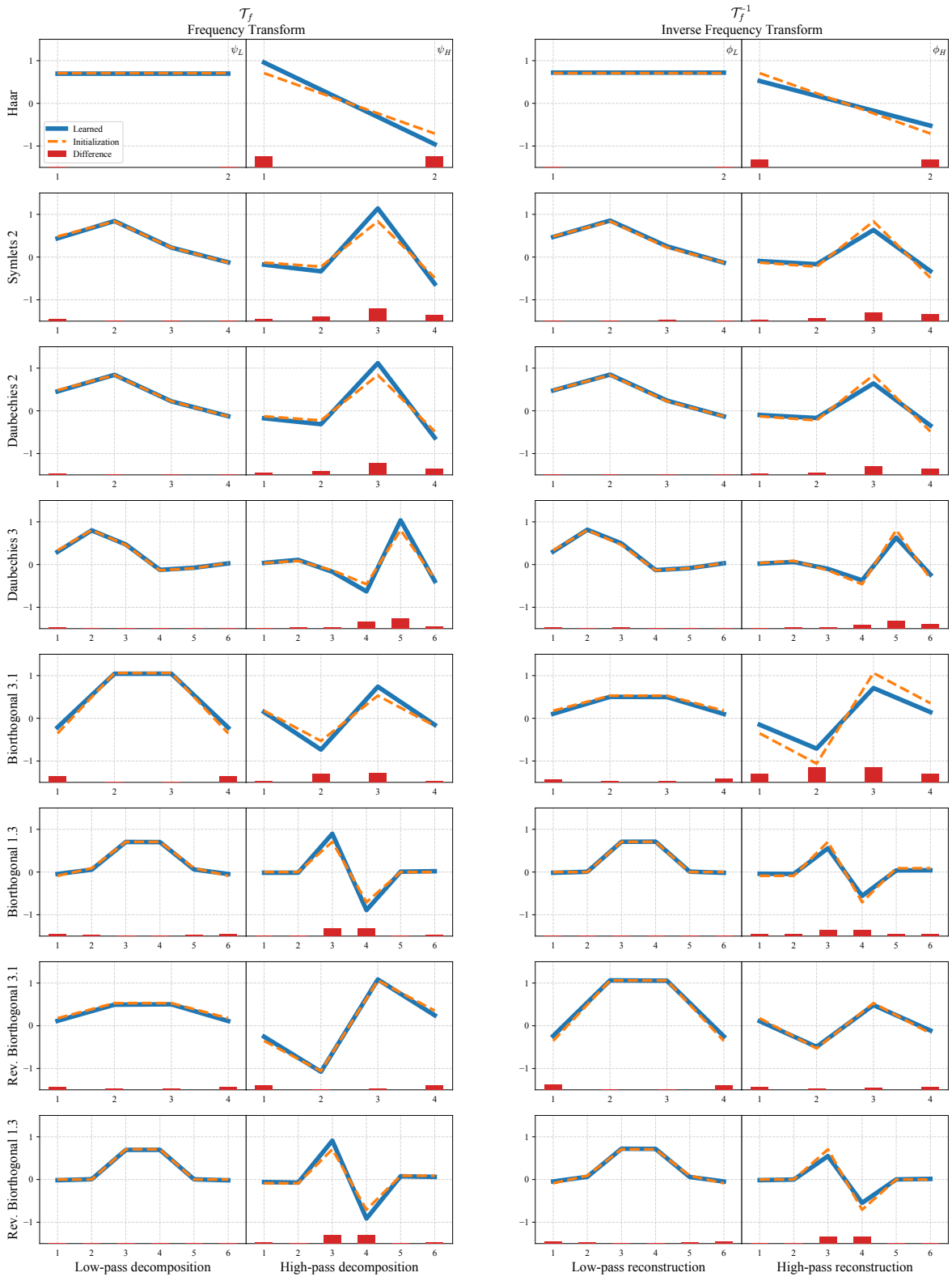


Figure 3: Learned frequency transform coefficients (blue) compared to the corresponding wavelet filters used as initialization (dashed orange). Red bar charts denote the difference between initial and learned coefficients. Refer to Section 2 for details.

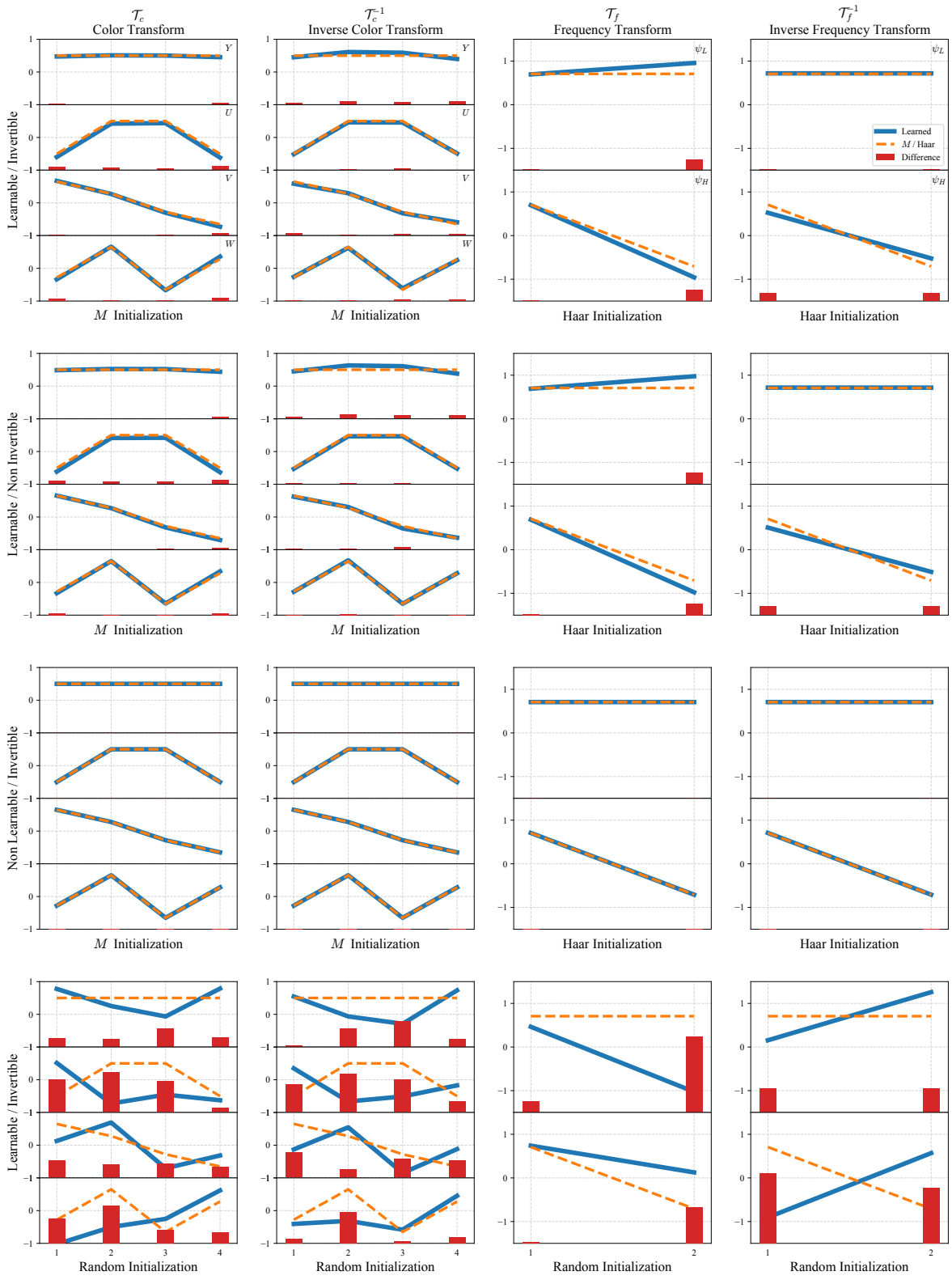
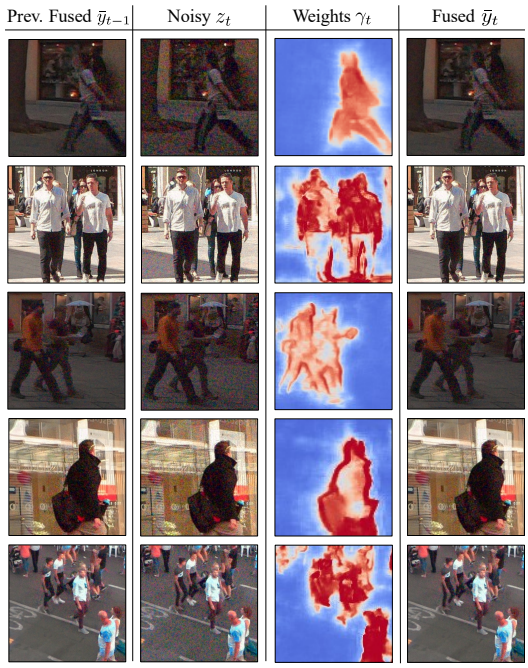
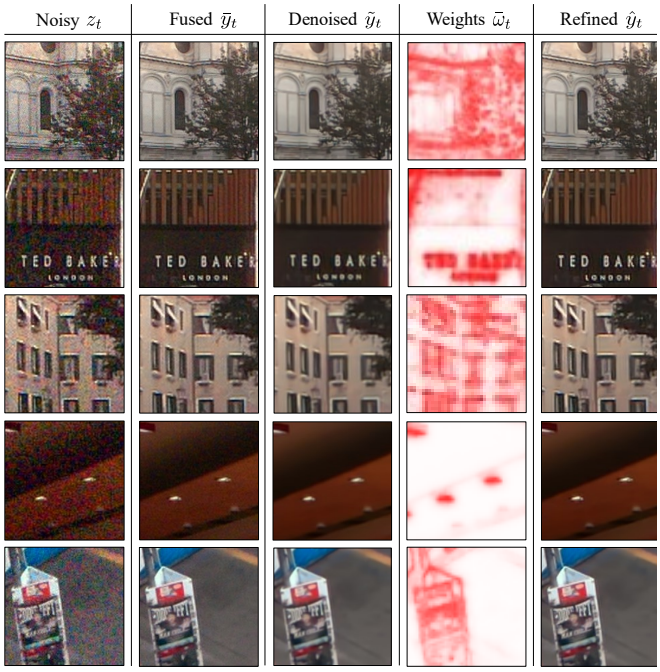


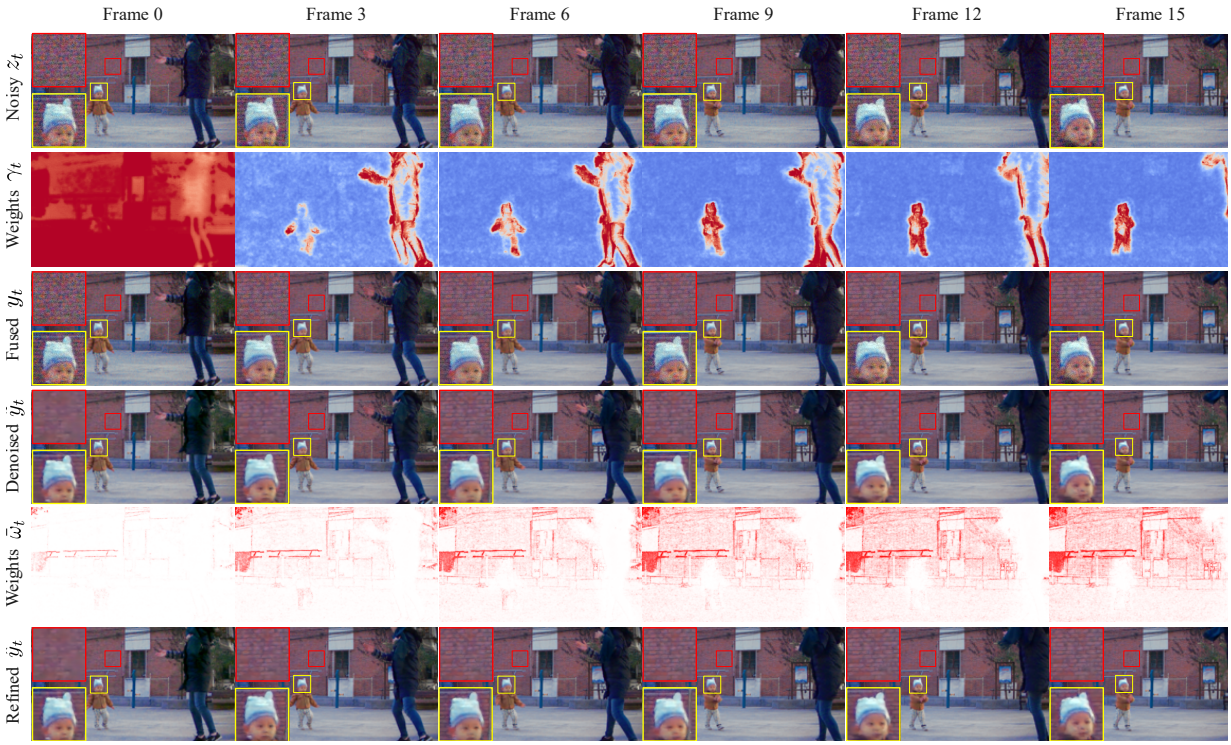
Figure 4: Learned coefficients for different initialization strategies and training settings. Red bar charts denote the difference between learned coefficients and initial color and Haar transforms. Refer to Section 2 for details.



(a) Examples of fusion stage.

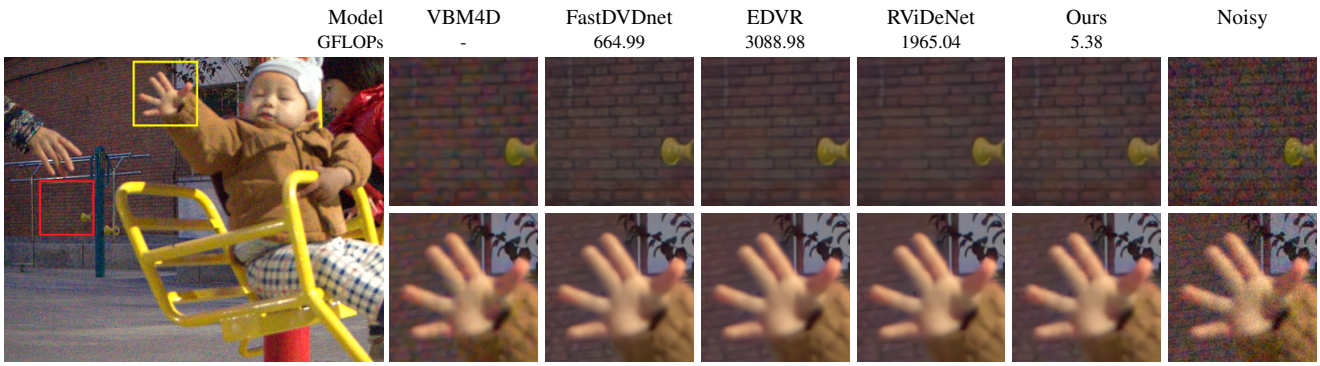


(b) Examples of fusion, denoising, and refinement stages.

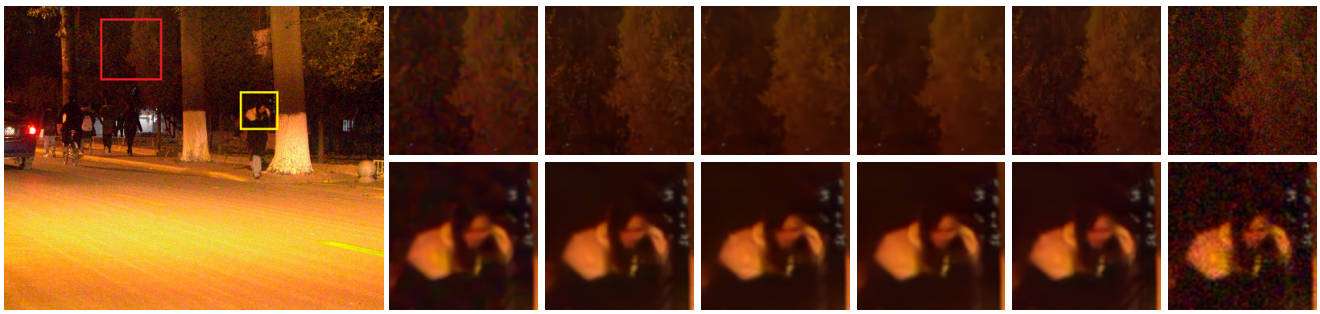


(c) Fusion, denoising, and refinement stages at different frames in one CRVD [8] sequence captured at ISO 25600.

Figure 5: Example of sequences processed by the proposed multi-stage EMVD. Refer to Section 2 for details.



(a) ISO 12800.



(b) ISO 12800.

Figure 6: Visual comparisons on different videos from the CRVD dataset [8].

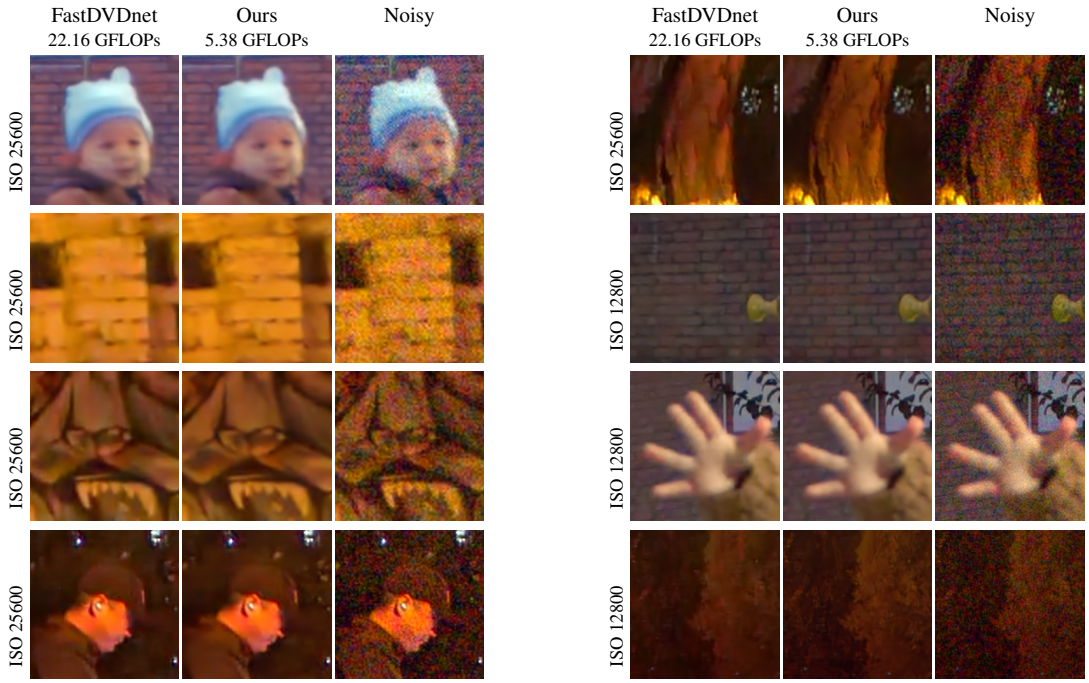


Figure 7: Visual comparisons of low-complexity models on different videos from the CRVD dataset [8].

References

[1] Lucio Azzari and Alessandro Foi. Gaussian-Cauchy mixture modeling for robust signal-dependent noise estimation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5357–5361, 2014. 4

[2] 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, 2020. 1

[3] 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. 4

[4] Victor Lempitsky, Andrea Vedaldi, and Dmitry Ulyanov. Deep image prior. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9446–9454, 2018. 2

[5] Matteo Maggioni, Giacomo Boracchi, Alessandro Foi, and Karen Egiazarian. Video denoising, deblocking, and enhancement through separable 4-D nonlocal spatiotemporal transforms. *IEEE Transactions on image processing*, 21(9):3952–3966, 2012. 1, 4

[6] Matias Tassano, Julie Delon, and Thomas Veit. FastDVDnet: Towards real-time deep video denoising without flow estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 4

[7] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. EDVR: Video restoration with enhanced deformable convolutional networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 0–0, 2019. 2, 4

[8] Huanjing Yue, Cong Cao, Lei Liao, Ronghe Chu, and Jingyu Yang. Supervised raw video denoising with a benchmark dataset on dynamic scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2301–2310, 2020. 1, 2, 4, 7, 8