Real-time Controllable Denoising for Image and Video

Zhaoyang Zhang[1*] Yitong Jiang[1*] Wenqi Shao[1,3] Xiaogang Wang[1] Ping Luo[2,3] Kaimo Lin[4] Jinwei Gu[1,3]
[1]The Chinese University of Hong Kong [2]The Univesity of Hong Kong [3]Shanghai AI Laboratory [4]SenseBrain
{zhaoyangzhang@link, ytjiang@link,weqish@link, xgwang@ee,jwgu@}.cuhk.edu.hk
linkaimo1990@gmail.com, pluo@cs.hku.hk

Abstract

Controllable image denoising aims to generate clean samples with human perceptual priors and balance sharpness and smoothness. In traditional filter-based denoising methods, this can be easily achieved by adjusting the filtering strength. However, for NN (Neural Network)-based models, adjusting the final denoising strength requires performing network inference each time, making it almost impossible for real-time user interaction. In this paper, we introduce Real-time Controllable Denoising (RCD), the first deep image and video denoising pipeline that provides a fully controllable user interface to edit arbitrary denoising levels in real-time with only one-time network inference. Unlike existing controllable denoising methods that require multiple denoisers and training stages, RCD replaces the last output layer (which usually outputs a single noise map) of an existing CNN-based model with a lightweight module that outputs multiple noise maps. We propose a novel Noise Decorrelation process to enforce the orthogonality of the noise feature maps, allowing arbitrary noise level control through noise map interpolation. This process is network-free and does not require network inference. Our experiments show that RCD can enable real-time editable image and video denoising for various existing heavy-weight models without sacrificing their original performance.

1. Introduction

Image and video denoising are fundamental problems in computational photography and computer vision. With the development of deep neural networks [12, 26, 49, 59], model-based denoising methods have achieved tremendous success in generating clean images and videos with superior denoising scores [4,55,57]. However, it should be noted that the improvement in reconstruction accuracy (e.g., PSNR, SSIM) is not always accompanied by an improvement in visual quality, which is known as the Perception-Distortion trade-off [6]. In traditional denoising approaches, we can easily adjust the denoising level by tuning related control parameters and deriving our preferred visual results. However, for typical deep network methods, we can only restore the degraded image or video to a fixed output with a predetermined restoration level.

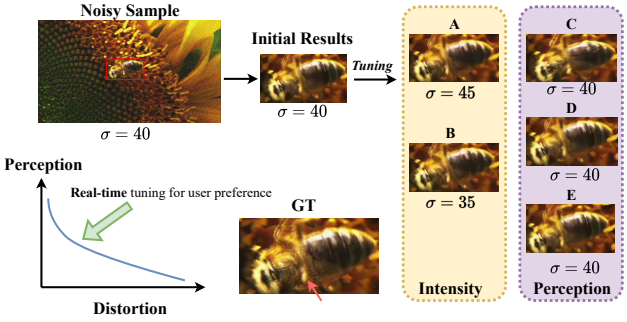

Figure 1. Real-time controllable denoising allows users further tuning the restored results to achieve Perception-Distortion trade-off. **A-B**: tuning with changing denoising intensity. **C-E**: tuning without changing denoising intensity.

In recent years, several modulation methods have been proposed to generate continuous restoration effects between two pre-defined denoising levels. These methods can be categorized into two kinds: interpolation-based methods [17,24,50,51], which use deep feature interpolation layers, and condition-network-based methods, which import an extra condition network for denoising control [9, 25, 39]. Essentially, both types of methods are designed based on the observation that the outputs of the network change continuously with the modulation of features/filters. This observation enables deep denoising control, but it also introduces several limitations. First, there is a **lack of explainability**, as the relationship between the control parameters (how to modulate features) and the control operation (how the network outputs are changed) is unclear [24]. This indicates that black-box operators (network layers) must be used to encode them. Second, the use of control parameters as network inputs requires entire network propagation each time control parameters change, resulting in a **lack of efficiency**. Lastly, current modulation methods often require an explicit degradation level during training, which is hard to obtain for **real-world samples**. As a result, current controllable denoising methods only focus on synthetic

*Equal contribution.

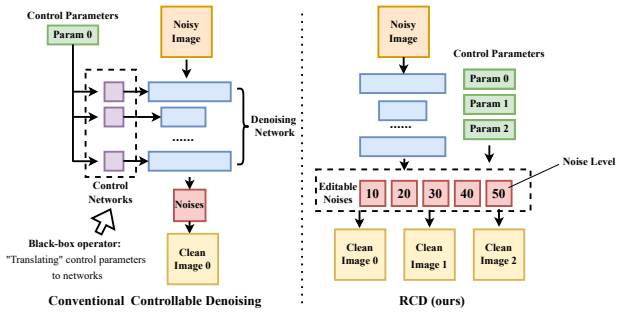

Figure 2. Comparison of pipelines between conventional controllable denoising and our RCD. RCD achieves real-time noise control by manipulating editable noises directly.

noise benchmarks. Furthermore, both interpolation-based and condition-network-based methods have their own drawbacks. Interpolation-based methods often require multiple training stages, including pretraining two basic models (start level and end level). On the other hand, condition-network-based methods are strenuous to jointly optimize the base network and the condition network.

In this paper, we research on the problem: Can we achieve real-time controllable denoising that abandons the auxiliary network and requires no network forward propagation for changing restoration effects at test time?

Towards this goal, we propose Real-time Controllable Denoising method (RCD), a lightweight pipeline for enabling rapid denoising control to achieve Perception-Distortion Balance (See Fig. 1). Our RCD can be plugged into any noise-generate-based restoration methods [11, 46, 54, 55] with just a few additional calculations. Specifically, we replace the last layer of an existing denoising network (which usually outputs a single noise map) with a lightweight module that generates multiple noise maps with different noise levels. We utilize a novel Noise Decorrelation process to enforce the orthogonality of the noise distribution of these noise maps during training. As a result, we can attain arbitrary denoising effects by simple linear interpolation of these noise maps. Since this process does not require network inference, it makes real-time user interaction possible even for heavy denoising networks.

Fig. 2 illustrates the fundamental differences between our RCD approach and conventional controllable denoising methods. In contrast to traditional methods that rely on control networks, the RCD pipeline generates editable noises of varying intensities/levels, providing explicit control by external parameters and enabling network-free, real-time denoising editing. Real-time editing capabilities offered by RCD create new opportunities for numerous applications that were previously impossible using conventional techniques, such as online video denoising editing, even during playback (e.g., mobile phone camera video quality tuning for ISP tuning engineers), as well as deploying controllable denoising on edge devices and embedded systems. Since the editing stage of RCD only involves image interpolation, users can edit their desired results on low-performance devices without the need for GPUs/DSPs.

Moreover, unlike previous methods that only support changing noise levels, RCD allows users to adjust denoising results at a specific noise level by providing a new interface to modify the noise generation strategy. RCD is also the first validated method for controllable denoising on real-world benchmarks. It is noteworthy that existing controllable methods typically require training data with fixed-level noise to establish their maximum and minimum noise levels, which makes them unsuitable for most real-world benchmarks comprising data with varying and unbalanced noise levels.

Our main contributions can be summarized as follows:

- We propose RCD, a controllable denoising pipeline that firstly supports **real-time denoising control** ($>$ **2000**$\times$ speedup compared to conventional controllable methods) and **larger control capacity** (more than just intensity) without multiple training stages [24] and auxiliary networks [50].
- RCD is the first method supporting controllable denoising on real-world benchmarks.
- We propose a general Noise Decorrelation technique to estimate editable noises.
- We achieve comparable or better results on widely-used real/synthetic image-denoising and video-denoising datasets with minimal additional computational cost.

2. Related Works

2.1. Denoising

Traditional image and video denoising methods are often based on prior assumptions such as sparse image prior [3, 15, 16, 20], non-local similarity [7, 13, 14, 18], and other similar techniques [22, 41, 52]. However, with the recent development of deep learning networks, many learning-based methods have been proposed and achieved state-of-the-art performance. Early works [8] utilized multi-layer perceptron (MLP) to achieve comparable results with BM3D. In recent years, there has been rapid progress on CNN-based denoising methods [4, 10, 21, 47, 55, 57] and Transformer-based methods [32, 42, 54, 59], which have started to dominate the image/video denoising task. However, the above-mentioned works mainly focus on designing novel network architectures to improve the denoising performance and usually generate a single output. Their lack of ability to adjust the output denoising level based on user's feedback has greatly restricted their practical use in many real-world applications. Moreover, although techniques like pruning [33, 38, 60] and quantization [45, 61] can accelerate such neural network-based methods, they are typically heavy, which restricts their application to real-time denoising control.

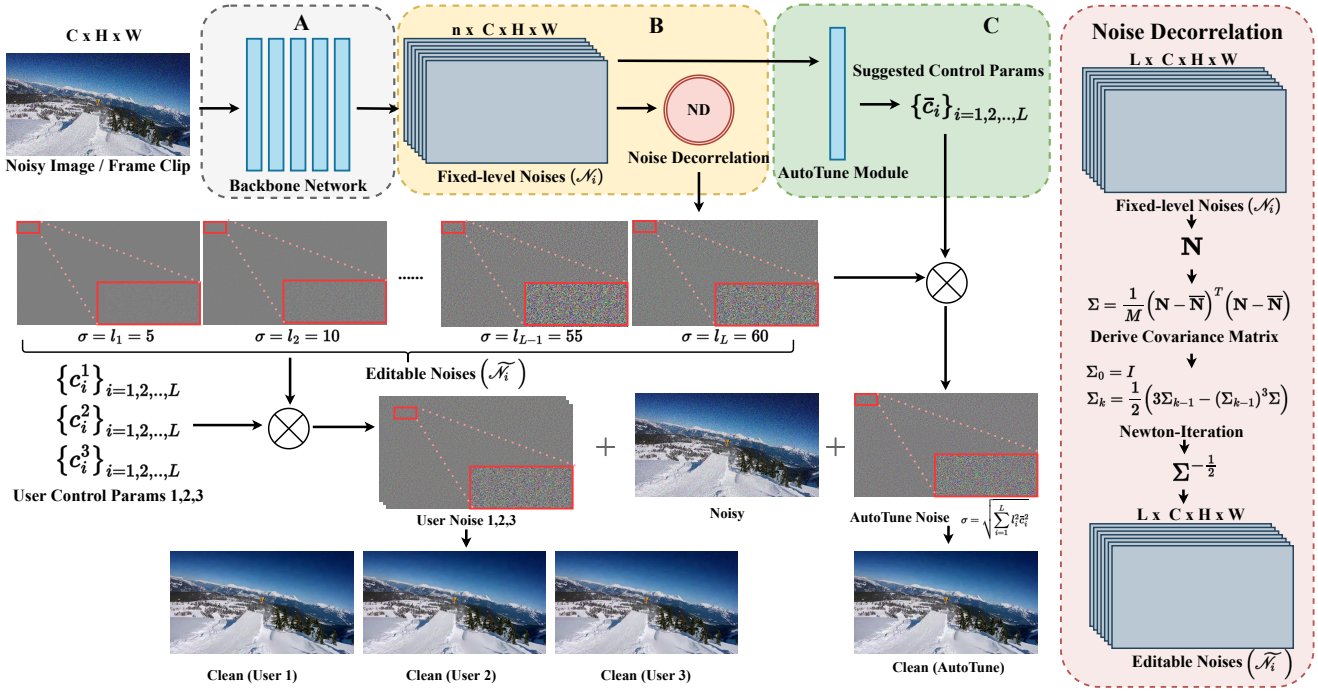

Figure 3. Pipeline overview of proposed RCD framework. **A:** Backbone network for generating multi-level noise maps. **B:** Noise Decorrelation module for editable noises. **C:** AutoTune module for providing reference control parameters for users.

2.2. Controllable denoising

Most conventional deep-learning methods for image/video denoising can only generate a fixed result with a specific restoration level. Recently, some controllable image/video denoising methods allow users to adjust the restoration effect without retraining the network. DNI [51] and AdaFM [24] used the observation that the learned filters of the models trained with different restoration levels are similar in visual patterns. DNI interpolated all the corresponding parameters between two related networks to derive smooth and continuous restoration effects, while AdaFM adopted feature modulation filters after each convolution layer. CFSNet [50] proposed an adaptive learning strategy of using interpolation coefficients to couple the intermediate features between the main branch and the tuning branch. Different from these interpolation-base methods, some other methods [9, 25, 39] regarded modulation as a conditional image restoration problem and adopted a joint training strategy. CUGAN [9] proposed a GAN-based image restoration framework to avoid the over-smooth problem, a common issue in PSNR-oriented methods. However, all of the above controllable methods can only be trained with synthetic degradations because they require explicit degradation levels during training. When applied on real-world data, as shown in [23], methods that trained for blind Additive White Gaussian Noise (AWGN) [35,55] may be overfitted and often suffer from dramatic performance drop. Besides the real-world image issue, all these controllable methods utilize an auxiliary conditional network and require one network inference for each different target restoration level at test time, which makes them almost impossible for real-time application.

3. Methods

3.1. Conventional Deep Denoising

Deep denoising methods trump traditional filter-based techniques by leveraging the neural networks' robust representation learning capability. Most current denoising methods [11, 32, 46] reason about the relationship between clean and noisy images by regressing noise maps with a neural generator. Specifically, given a noisy image $\mathbf{I_n}$ and model $\mathcal{M} : \mathbb{R}^{H\times W\times C} \rightarrow \mathbb{R}^{H\times W\times C}$, we can derive the predicted clean image $\mathbf{I_c}$ by: $\mathbf{I_c} = \mathbf{I_n} + \mathcal{M}(\mathbf{I_n})$, where model $\mathcal{M}$ is updated by minimizing the distance between the denoising result $\mathbf{I_c}$ and the ground truth $\mathbf{I_{gt}}$. As we can see, this kind of approach generates a single fixed output result in a black-box manner, making it almost impossible to adjust the denoise operation explicitly.

3.2. Pipeline Overview

In this section, we present Real-time Controllable Denoising (RCD), a novel deep learning-based pipeline for real-time controllable denoising. As illustrated in Fig. 3, RCD essentially consists of three parts: (1) A backbone network, *i.e.*, $\mathcal{M}_b : \mathbb{R}^{H\times W\times C} \rightarrow \mathbb{R}^{H\times W\times LC}$, generates multiple fixed-level noise maps, where L is the number of pre-defined noise levels (see (A) in Fig. 3). (2) A Noise Decorrelation (ND) block that enforces the editability of the generated noise maps (see (B) in Fig. 3). (3) An AutoTune

module that gives a default set of control parameters to generate the best denoising result.

Specifically, the backbone network will generate multiple fixed-level noise maps, *i.e.*, $\{\mathcal{N}_i\}_{i=1}^{L}$, for each noisy image input. The noise maps are then fed into the proposed Noise Decorrelation (ND) block, which makes noise maps orthogonal to each other. In this way, the decorrelated noise maps $\{\tilde{\mathcal{N}}_i\}_{i=1}^{L}$ will be zero-correlated and thus become linearly interpolable. At last, the AutoTune module will give a set of suggested control parameters $\{\bar{c}_i\}_{i=1}^{L}$ to generate the final denoising result as follows:

$$\mathbf{I_c} = \mathbf{I_n} + \sum_{i=1}^{L} \bar{c}_i \tilde{\mathcal{N}}_i, \tag{1}$$

where $\sum_{i=1}^{L} \bar{c}_i = 1$. Moreover, given the zero-correlated noise maps, users can also generate arbitrary strength denoising results by replacing $\{\bar{c}_i\}_{i=1}^{L}$ with their own customized control parameters $\{c_i\}_{i=1}^{L}$.

3.3. Multi-level Noise Generation

Given a noisy input image $\mathbf{I_n}$, the backbone network aims to generate multiple noise maps $\{\mathcal{N}_i\}_{i=1}^{L}$, corresponding to a set of pre-defined noise levels $\{l_i\}_{i=1}^{L}$, *e.g.* noise levels $\{5, 10, 15, ..., 60\}$. Hence, we have

$$\sigma(\mathcal{N}_i) = l_i, \forall i = 1, ..., L, \tag{2}$$

where σ is the noise level operation that calculates the standard deviation of pixels in each noise map. To obtain multi-level noise maps, we replace the conventional last output layer of the denoising network with a convolutional layer with an output channel size of $L \cdot C$. Moreover, the level of the noise map is explicitly generated with the normalization operation, as given by

$$\mathcal{N}_i = l_i \frac{\mathcal{M}_b(\mathbf{I_n})^{(i)}}{\sigma(\mathcal{M}_b(\mathbf{I_n})^{(i)})}, \forall i = 1, ..., L. \tag{3}$$

Here $\mathcal{M}_b(\mathbf{I_n}) \in \mathbb{R}^{H \times W \times LC}$ is network output, and $\mathcal{M}_b(\mathbf{I_n})^{(i)} \in \mathbb{R}^{H \times W \times C}$ is the i-th component separated from the channel dimension. The derived $\mathcal{N}_i$ can be considered as the noise map estimated at the given noise level l_i.

Different from prior controllable denoising methods with implicit interpolation in the network, we propose to explicitly interpolate the noise maps in Eqn. 3. Thanks to the separation of noise interpolation and network inference, our RCD can achieve real-time user interaction.

However, the multi-level noise maps $\mathcal{N}_i$ directly obtained by convolutional layers are usually highly correlated, which leads to the problem of noise level collapse. In other words, the noise map representations in different levels are redundant, implying that the number of noise maps at different noise strengths that participate in the linear interpolation in Eqn. 1 is implicitly reduced. Without any constraint,

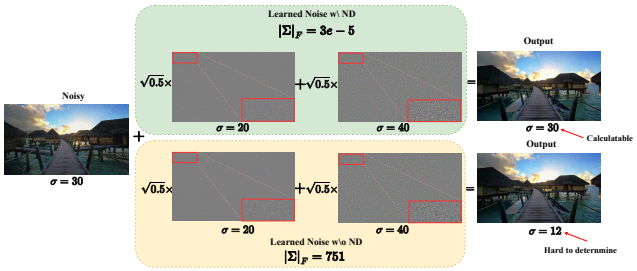

Figure 4. Demonstration of Noise Decorrelation's influence on noise editing. $|\Sigma|_F$ denotes norms of the covariance matrix for corresponding learned noises and σ is noise intensity.

our experiments show that the single noise map at a certain noise level would dominate in the linear interpolation for a variety of input noisy images. To address this issue, we further introduce the Noise Decorrelation block to make representations of these noise maps much more informative in the following section.

3.4. Noise Decorrelation

The Noise Decorrelation (ND) block is designed to regularize the backbone network to generate editable noise maps at varying levels. In particular, this block is a parameter-free computational unit that enforces $\mathcal{N}_i$ to be approximately zero-correlated with each other: $\mathbf{cov}(\mathcal{N}_i, \mathcal{N}_j) \approx 0, \forall i, j \in \{1, 2, ..., L\}$, where $\mathbf{cov}(\cdot, \cdot)$ is covariance operator. Inspired by the success of using the decorrelation technique in network optimization and normalization, we adopt the whitening-based methods [29, 30] for noise decorrelation here. The noise maps are decorrelated using the inverse square root of their covariance matrix.

Specifically, for each predicted fixed-level noise map $\mathcal{N}_i$, it will firstly be reshaped to $\mathcal{N}_i \in \mathbb{R}^{1 \times M}$, where $M = HWC$. By stacking the reshaped $\mathcal{N}_i$ over the first dimension, we have $\mathbf{N} \in \mathbb{R}^{L \times M}$. We then calculate the noise covariance matrix σ by: $\Sigma = \frac{1}{M-1}(\mathbf{N} - \bar{\mathbf{N}})(\mathbf{N} - \bar{\mathbf{N}})^T$ where $\bar{\mathbf{N}}$ is mean of $\mathbf{N}$ over channel M.

The Noise Decorrelation block needs to compute inverse square root $\Sigma^{-\frac{1}{2}} \in \mathbb{R}^{L \times L}$, which can be done by eigen decomposition or SVD. Since this kind of operation involves heavy computation [29], we instead adopt the more efficient Newton's Iteration to estimate $\Sigma^{-\frac{1}{2}}$ as in [5, 27]. Giving a covariance matrix Σ, Newton's Iteration calculates $\Sigma^{-\frac{1}{2}}$ by following the iterations below:

$$\begin{aligned} &\Sigma_0 = I, \\ &\Sigma_k = \tfrac{1}{2}(3\Sigma_{k-1} - (\Sigma_{k-1})^3\Sigma), k = 1, 2, .., T, \end{aligned} \tag{4}$$

where k is the iteration index and T is the iteration number (in our experiments T = 3 or 4). Σ_k is guaranteed to converge to $\Sigma^{-\frac{1}{2}}$, if $\|I - \Sigma\|_2 < 1$ [5]. This condition can be achieved by normalizing Σ to $\frac{\Sigma}{tr(\Sigma)}$, where $tr(.)$ is trace operator.

The derived $\Sigma^{-\frac{1}{2}}$ can be regarded as a whitening matrix [43], which decorrelates the noise maps $\mathbf{N}$ in

a differentiable manner. The decorrelated noise maps $\mathbf{N} \in \mathbb{R}^{\tilde{H} \times W \times LC}$ can be obtained by calculating: $\tilde{\mathbf{N}} = \Sigma^{-\frac{1}{2}}\mathbf{N}$. We can then have our editable fixed-level noises $\tilde{\mathcal{N}}_i \in \mathbb{R}^{H \times W \times C}$ by reshaping $\tilde{\mathbf{N}}$ and splitting it into L noise maps. After Noise Decorrelation, we apply the same normalization as Eqn. 3 to guarantee the noise strength of the decorrelated noises.

The zero-correlated noise maps $\tilde{\mathcal{N}}_i$ present several excellent properties for controllable denoising. Firstly, the linearity of the noise level's square towards $\tilde{\mathcal{N}}_i$ is guaranteed. In other words, given an arbitrary set of control parameters $\{c_i\}_{i=1}^{L}$, we have

$$\mathbf{Var}(\sum_{i=1}^{L} c_i\tilde{\mathcal{N}}_i) = \sum_{i=1}^{L} c_i^2\mathbf{Var}(\tilde{\mathcal{N}}_i), \tag{5}$$

where $\mathbf{Var}(\cdot)$ denotes variance operator. Apparently, Eqn. 5 holds when elements of $\{\tilde{\mathcal{N}}_i\}_{i=1,2,..,m}$ are mutually zero-correlated. Eqn. 5 reveals the explicit relationship between the control parameters and the target noise level, which allows us to directly edit noises by interpolating $\mathcal{N}_i$ using c_i. Secondly, the Noise Decorrelation block can be regarded as a regularization tool that forces models to learn different noise formats for each level, which will increase the representation capacity of the denoising network [30].

Fig. 4 demonstrates how the Noise Decorrelation block works. With ND block, the covariance of learned noises is reduced to almost zero (without it, $|\Sigma|_F$ can be 751 and unignorable), allowing us to derive determined interpolated results with target noise intensity. In contrast, without the Noise Decorrelation blocks, the output noise level can not be guaranteed.

3.5. AutoTune Module

Given the decorrelated noise maps from the Noise Decorrelation block, the AutoTune module will predict a set of model-suggested control parameters, *i.e.*, $\{\bar{c}_i\}_{i=1}^{L}$, to generate the default denoising result. Users can then use this set of parameters as a starting point to fine-tune their final desired denoising strength. Our AutoTune module is extremely lightweight, and is formulated as a single-layer module with temperature softmax activation. Specifically, $\{\bar{c}_i\}_{i=1}^{L}$ can be obtained by : $\bar{c}_i = \frac{e^{\frac{\mathcal{A}(f)_i}{\tau}}}{\sum_{j=1}^{n} e^{\frac{\mathcal{A}(f)_j}{\tau}}}$, where $\mathcal{A}$ is the NN layer, f is the input feature maps, and τ is temperature. In our experiments, τ is set to be 0.05 for best performance. Following the design ethos of efficiency and least coupling to the backbone architecture, we directly choose the unnormalized model outputs $\mathcal{M}(\mathbf{I_n})$ as f (see (C) in Fig. 3).

3.6. New Cardinality for Denoising Control.

Unlike existing methods that only modulate noise intensity, our RCD control scheme allows users to further optimize the denoising result to a given noise intensity by tuning $\{c_i\}$, as long as the weighted mean of l_i towards c_i remains the same. Eqn. 5 shows that when $\sum_{i=1}^{L} c_i^2\mathbf{Var}(\tilde{\mathcal{N}}_i) = \sum_{i=1}^{L} c_i^2 l_i^2$ is fixed, the variance of the output noise $\mathbf{Var}(\sum_{i=1}^{L} c_i\tilde{\mathcal{N}}_i)$ would be also fixed. By exploring c_i under the condition of fixed $\sum_{i=1}^{L} c_i^2 l_i^2$, we can further optimize the denoising results at a specific noise level by involving different components ($\mathcal{N}_i$) in the noise interpolation. As shown in Fig. 5, our AutoTune module can generate high-quality results using just the reference control parameters $\{\bar{c}_i\}$. Users can further improve the result by artificially tuning $\{c_i\}$ around $\{\bar{c}_i\}$, even at the same noise level.

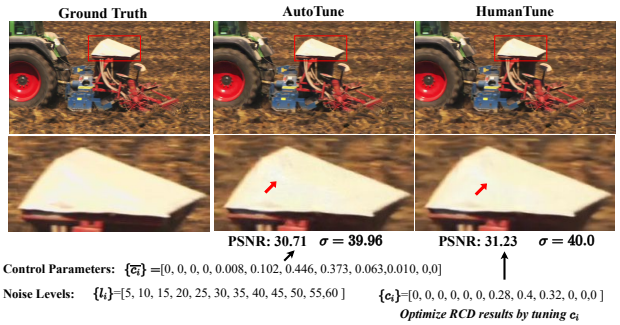

Figure 5. Example of RCD denoising results by AutoTune and HumanTune on Set8. AutoTune module provides reference control parameters, *i.e.*, $\{\bar{c}_i\}_{i=1}^{L}$), to generate the denoising result, and it can be further improved by fine-grained artificial tuning (HumanTune), *i.e.*, $\{c_i\}$, without changing the noise intensity (both $\sigma = 40$).

3.7. Optimization Targets

To guarantee the visual quality of arbitrarily edited noise $\sum_{i=1}^{L} c_i\tilde{\mathcal{N}}_i$ for any given set of control parameters $\{c_i\}_{i=1}^{L}$, we adopt a multi-level concurrent training strategy by minimizing the difference between each level's noise output and the ground truth noise, *i.e.*, $\mathcal{L}_{level}$, which is derived by:

$$\mathcal{L}_{level} = \frac{1}{L}\sum_{i=1}^{L} \mathcal{L}(\mathbf{I}_{gt}, \mathbf{I_n} + \tilde{\mathcal{N}}_i) \tag{6}$$

where $\mathbf{I}_{gt}$, $\mathbf{I_n}$ are ground truth clean image and the input noisy image. $\mathcal{L}(.)$ can be any loss functions (e.g., L2 loss or PSNR loss).

Spectacularly, $\mathbf{I_n} + \tilde{\mathcal{N}}_i$ can be regarded as corner cases of RCD when we use one-hot control parameters as input. Joint optimization of all the noise levels ensures that each element of $\mathcal{N}_i$ can be trained as optimal noise estimation under the condition of fixed noise level l_i.

Together with the AutoTune module optimization, our final cost function can be written as:

$$\mathcal{L}_{total} = \lambda\mathcal{L}_{level} + \mathcal{L}(\mathbf{I}_{gt}, \mathbf{I_n} + \sum_{i=1}^{L} \bar{c}_i\tilde{\mathcal{N}}_i), \tag{7}$$

Table 1. Gaussian single image denoising results (PSNR). RCD is evaluated with AutoTune results. "-": not reported

Method	Controllable	CBSD68			Kodak24			McMaster			Urban100		
		$\sigma = 15$	$\sigma = 25$	$\sigma = 50$	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$
IRCNN [56]	✗	33.86	31.16	27.86	34.69	32.18	28.93	34.58	32.18	28.91	33.78	31.20	27.70
FFDNet [57]	✗	33.87	31.21	27.96	34.63	32.13	28.98	34.66	32.35	29.18	33.83	31.40	28.05
DnCNN [55]	✗	33.90	31.24	27.95	34.60	32.14	28.95	33.45	31.52	28.62	32.98	30.81	27.59
DSNet [40]	✗	33.91	31.28	28.05	34.63	32.16	29.05	34.67	32.40	29.28		-	
CResMD [25]	✓	33.97	-	28.06		-			-			-	
AdaFM-Net [24]	✓	34.10	31.43	28.13		-			-			-	
NAFNet [11]	✗	34.11	31.49	28.27	35.14	32.70	29.68	35.07	32.82	29.79	34.41	32.09	29.00
NAFNet-RCD (ours)	✓	**34.13**	**31.49**	**28.26**	**35.15**	**32.72**	**29.69**	**35.11**	**32.84**	**29.81**	**34.45**	**32.12**	**29.02**

Table 2. Running time comparison for RCD and other controllable methods during test time. **Full Pipeline** compares full pipeline latency for the model to infer 1000 images, and **Edit-only** compares latency for editing one image with 1000 different control parameters.

Method	Multi-stage training	Full Pipeline	Edit-only
AdaFM-Net	required	81.03s	81.03s
CResMDNet	not required	128.08	128.08s
NAFNet-RCD	not required	**64.84s**	**0.04s**

where λ is the loss weight ($\lambda = 0.1$ in our experiments), and $\mathcal{L}(\mathbf{I}_{gt}, \mathbf{I_n} + \sum_{i=1}^{n} \bar{c}_i \tilde{\mathcal{N}}_i)$ optimizes the denoising result derived by the model-suggested control parameters.

4. Experiments

This section is organized as follows: First, we demonstrate the effectiveness of our plug-in RCD with SOTA image denoising methods [11] in different scales on synthetic noise datasets. Next, to evaluate the ability of blind denoising on real-world data, we conduct experiments on popular real-world denoising dataset SIDD [1]. Then, we apply our real-time controllable RCD pipeline on video denoising applications. At last, we empirically discuss some design details described in the previous sections.

4.1. Gaussian Single Image Denoising

Experimental Setup. To fully demonstrate the effectiveness of the proposed RCD, we choose the most recent SOTA method NAFNet [11] as our backbone. Following [54], we first conduct denoising experiments on several widely-used synthetic color image benchmarks (DIV2K [2], BSD400 [36], Flickr2K [53] and WaterlooED [34]) with additive white Gaussian noise ($\sigma \in [0, 60]$). The training patch size is 128 × 128 and the batch size is 64. We train our model with Adam [31] optimizer and learning rate $1e-3$ for total 60K iterations. Consistent to [11], PSNR loss is adapted as the loss function. Both the baseline model (NAFNet) and its RCD variants (NAFNet-RCD) are trained from scratch. For settings of RCD, we initialize $L = 12$ and $\{l_i\} = [5, 10, ..., 60]$ for synthetic denoising training.

Complexity analysis. Extensive adjustments of controllable parameters are often required to obtain one satisfying result for users. Therefore, editing time is vital for controllable methods. This section compares the inference and editing latency of our RCD and conventional controllable pipelines on GTX 1080Ti. As shown in Tab. 2, the proposed RCD not only outperforms other conventional controllable pipelines on inference time, but more importantly, can overwhelm those traditional controllable designs on editing time, which can be more than **2000** times faster (as editing process of RCD is network-free, without reliance on sub-networks). This comparison confirms that our RCD is more than enough for real-time image editing.

Results Analysis. We evaluate our proposed method on widely used synthetic noise datasets CBSD68 [37], Kodak24 [19], McMaster [58] and Urban100 [28] with noise levels $\sigma(15)$, $\sigma(15)$ and $\sigma(50)$. RCD is evaluated with denoising results using AutoTune outputs$\{\bar{c}_i\}$. As shown in Tab. 1, NFANet-RCD achieves comparable performance to the baseline NFANet consistently on multiple datasets, indicating that our plug-in RCD module enables real-time controllable denoising for NAFNet without sacrificing its original denoising performance. Please note that NAFNet-RCD can yield comparable results to the backbone just by using the AutoTune outputs, and the performance can be further improved by manually tuning the control parameters (See Sec. 3.6.) We further show the qualitative performance of NAFNet-RCD in Fig. 6. NAFNet-RCD can recover more details of some degraded images, which may be benefited from RCD's richer representation capacity by integrating multiple noise maps.

Slimmer Model Variants. Towards the goal of evaluating the compatibility and robustness of RCD, we conduct ablations by applying RCD to different-sized backbones. Specifically, we shrink the width and block numbers of NAFNet, denoting derived models as NAFNet-small ($\frac{1}{4}\times$) and NAFNet-tiny ($\frac{1}{16}\times$). Tab. 3 reports the results of RCD with those scaled backbones. It can be observed that the RCD-variants can achieve comparable and even slightly better denoising results compared to their baselines, which further demonstrates RCD's robustness and effectiveness for different-sized backbones.

4.2. Real Single Image Denoising

Experimental Setup (Real Image) Unlike existing controllable denoising methods [24,50] which focus on synthetic benchmarks, we are the first solution that attempts to extend controllable denoising to real-world SIDD datasets.

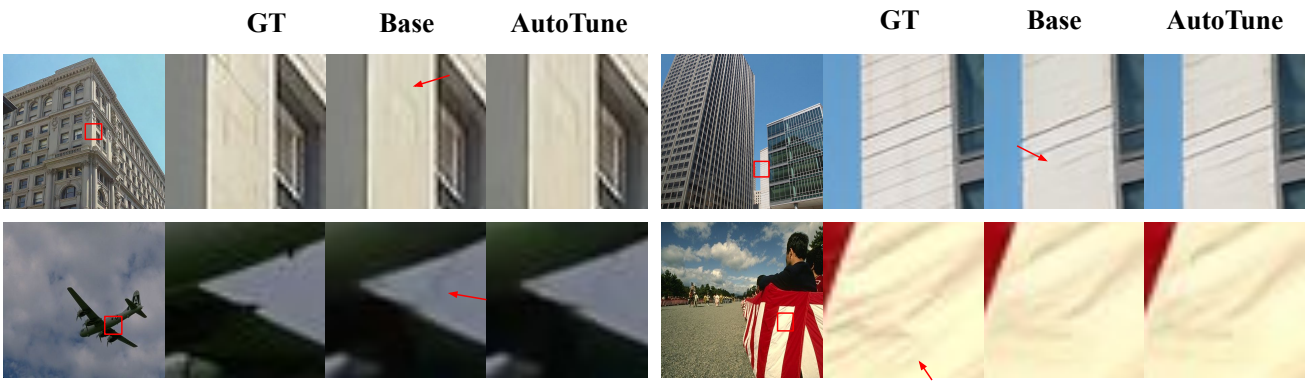

Figure 6. Visual comparison of RCD and their baseline results on $\sigma = 50$ denoising. **GT**: Ground truth. **Base**: Baseline model without RCD. **AutoTune**: RCD results by applying control parameters from AutoTune module.

Table 3. Ablation of RCD on various backbone sizes.

Method	CBSD68			Kodak24			McMaster			Urban100		
	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$
NAFNet-tiny	33.58	30.91	27.62	34.33	31.84	28.63	33.85	31.61	28.55	32.96	30.37	26.92
NAFNet-RCD-tiny	**33.71**	**31.06**	**27.68**	**34.46**	**31.98**	**28.65**	**34.07**	**31.78**	**28.61**	**33.22**	**30.66**	**27.18**
NAFNet-small	33.84	31.18	27.91	34.68	32.18	29.01	34.68	32.18	29.01	33.61	31.10	27.68
NAFNet-RCD-small	**33.96**	**31.31**	**28.05**	**34.83**	**32.32**	**29.14**	**34.71**	**32.40**	**29.26**	**33.92**	**31.46**	**28.08**
NAFNet	34.11	31.49	28.27	35.14	32.70	29.68	35.07	32.82	29.79	34.41	32.09	29.00
NAFNet-RCD	**34.13**	**31.49**	**28.26**	**35.15**	**32.72**	**29.69**	**35.11**	**32.84**	**29.81**	**34.45**	**32.12**	**29.02**

Table 4. Image denoising results on SIDD. **Real noise**: results on real-world SIDD test sets. **Synthetic noise**: results on SIDD test set with additive Gaussian noise ($\sigma = 25$).

Method	Real noise		Synthetic noise	
	PSNR	SSIM	PSNR	SSIM
NAFNet-tiny	**42.19**	**0.9796**	38.46	0.9551
NAFNet-RCD-tiny	41.86	0.9781	**38.60**	**0.9558**
NAFNet	**43.22**	**0.9818**	38.85	0.9481
NAFNet-RCD	42.91	0.9806	**39.14**	**0.9580**

SIDD consists of real noisy images captured by smartphones with $\sigma \in [0, 50]$. Instead of using full SIDD data, we choose subsets of SIDD with $\sigma \in [0, 12]$ (around 70% of the entire dataset) to train our RCD model, which is initialized with $L = 4$ and $\{l_i\} = [3, 6, 9, 12]$. The main reason is the lacking of high σ data at given levels in SIDD because of SIDD's highly long-tailed noise level distribution. Specifically, most noisy images in SIDD gather in $\sigma < 12$ and the samples distribute sparsely when σ is large. Consistent to Sec. 4.1, we adopt NAFNet (SOTA methods for SIDD challenge [11]) as our backbone at two scales ($1\times$, $\frac{1}{16}\times$). Both NAFNet-RCD and the corresponding baselines are trained on this subset with the same training settings as in [11].

Results and Analysis We conduct blind denoising experiments on SIDD with different RCD model scales to evaluate its adjustability to the real-world dataset. As shown in Tab. 4 (left), our RCD (AutoTune results) can achieve high-quality controllable real-world denoising in both model scales. However, we note that enabling controllable denoising with RCD may still result in a slight decrease in quantitative results (about 0.3dB), which may be a result of unbalanced data for each level and short noise level interval ($|l_{i+1} - l_i|$, see more discussion in Sec. 4.4).

SIDD with synthetic noise. We extensively conduct synthetic denoising experiments on SIDD to further show the compatibility of RCD on SIDD datasets. Following Sec. 4.1, we add random Gaussian noise $\sigma \in [0, 60]$ to SIDD training data, and both methods are evaluated on $\sigma = 50$ SIDD test samples. As shown in Tab. 4 (right), RCD models slightly outperform their baselines, demonstrating RCD's compatibility for SIDD. Moreover, this result can also indicate that RCD's performance drop on SIDD real image may arise from the noise distribution and RCD configurations, rather than RCD's adaptive capacity to SIDD data. See Appendix for more results and visualizations.

4.3. Video Denoising

Experiment Setup Following common practice [32, 44, 46], we train our models on DAVIS training set and use DAVIS-test and Set-8 for benchmarking. Like in [46], we add Gaussian noise with random standard deviation between 5-50 on the DAVIS clean videos for training. The DAVIS set contains 30 color sequences of resolution 854×480, which will be randomly cropped into 128×128 patches during training. Other training settings and hyperparameters are kept the same as [46] for a fair comparison.

Choice of Basic model. We choose FastDVD [46] as

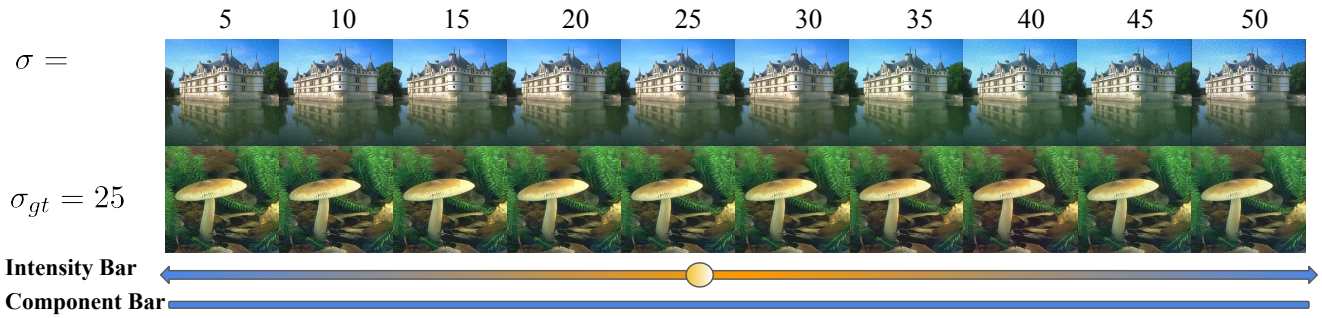

Figure 7. Illustration of RCD control logics. Users can retouch the **denoising level** by tuning the **Intensity** bar ($\sigma = \sqrt{\sum_{i=1}^{L} c_i^2 l_i^2}$) and setup their perceptual preference at **fixed level** by tuning **Component** bar (changing $\{c_i\}$ while keeping σ).

our backbone model. Although recent methods [32,48] outperform FastDVD by 1-2 PSNR at most, they actually introduce huge models and extra heavy operations like patch clustering [48] and layer-wise frame-to-frame wrapping using optical flow [32] ($> 100\times$ slower than FastDVD).

Results and Analysis. Like [44], we evaluate our video denoising models with the input length of one frame and five frames. We denote RCD models for video denoising as "FastDVD-RC" and compare their quantitative AutoTune denoising results to baseline FastDVD in Tab. 5. Consistent with preceding sections, AutoTune results of FastDVD-RCD can demonstrate comparable performance to the default FastDVD, which means our RCD can also achieve lossless real-time noise editing in video scenarios. Unlike previous heavy controllable denoising methods, our real-time RCD can even allow users to do online video denoising editing without any latency.

Table 5. Video denoising results.

Test set	σ	1 frame		5 frames	
		FastDVD	FastDVD-RCD	FastDVD	FastDVD-RCD
DAVIS	20	34.17	**34.21**	**35.69**	35.65
	30	32.45	**32.69**	**34.06**	34.04
	40	31.39	**31.60**	**32.80**	32.78
	50	30.26	**30.57**	31.83	**31.85**
Set 8	20	31.99	**32.01**	33.43	**33.46**
	30	30.61	**30.65**	31.62	**31.71**
	40	29.62	**29.83**	30.36	**30.42**
	50	28.61	**28.85**	29.41	**29.60**

4.4. Discussions

Selection of Denoising Levels. Differing from conventional denoising methods, RCD requires a group of predefined noise levels $\{l_i\}_{i=1}^{L}$. To evaluate how the selection of $\{l_i\}_{i=1}^{L}$ affects RCD's performance, we conduct ablation studies on FastDVD-RCD by changing the number of noise maps (L) (See Tab. 6.) All of the models are trained on noisy images with $\sigma \in (0, 60]$ and uniformly sampled noise levels that $\{l_i = \frac{60}{L} * i\}_{i=1}^{L}$. We observe that larger L means more fine-grained control on denoising, but it may incur a performance drop. In fact, when n is large we find that $\mathcal{L}_{level}$ will also keep large and be hard to optimize. Trading-off performance and control precision, we empirically choose $L = 12$ and noise level interval $|l_{i+1} - l_i| = 5$ as defaults.

Table 6. Ablations of FastDVD-RCD on different number of noise levels. Reported scores are PSNR of AutoTune outputs and GT.

Test Set	σ	$L = 1$	$L = 2$	$L = 12$	$L = 30$	$L = 60$
Set8	20	31.87	31.42	**32.01**	31.39	31.07
	30	30.51	30.09	**30.65**	30.12	29.75
	40	29.60	29.31	**29.83**	29.33	29.01
	50	28.62	28.29	**28.85**	28.22	28.05

Control Capacity. This section discusses the representation capacity of c_i as control parameters. Generally, c_i controls the denoising process on two aspects: intensity and components. Firstly, the noise levels of RCD outputs are identical and can be derived by $\sigma = \sqrt{\sum_{i=1}^{L} c_i^2 l_i^2}$ (See Sec. 3.4), which allows us to control the denoising intensity by changing $\{c_i\}$. Fig. 7 depicts visualizations of RCD-controlled denoising under different intensity settings. Besides, as discussed in Sec. 3.6, RCD supports further optimization of the denoising results at specific noise intensity by tuning c_i by involving different components of $\tilde{\mathcal{N}}_i$. (Please be reminded that $\tilde{\mathcal{N}}_i$ is trained by $\mathcal{L}_{level}$, denoting learned optimal denoising results at every fixed level l_i.)

5. Summary

We present RCD framework that enables real-time noise editing for controllable denoising. Unlike existing continual-level denoising methods, RCD doesn't require multiple training stages and auxiliary networks. With the proposed Noise Decorrelation module, RCD transforms the control of denoising into white-box operations, with no requirement to feed control parameters to networks at test time, which enables real-time editing even for heavy network models. Extensive experiments on widely-used real/synthetic image and video denoising datasets demonstrate the robustness and effectiveness of our RCD.

6. Acknowledgement

This paper is partially supported by the National Key R&D Program of China No.2022ZD0161000, the General Research Fund of HK No.17200622, and Shanghai Committee of Science and Technology (Grant No. 21DZ1100100).

References

[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown. A high-quality denoising dataset for smartphone cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1692–1700, 2018. 6

[2] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 126–135, 2017. 6

[3] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11):4311–4322, 2006. 2

[4] Saeed Anwar and Nick Barnes. Real image denoising with feature attention. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3155–3164, 2019. 1, 2

[5] Dario A Bini, Nicholas J Higham, and Beatrice Meini. Algorithms for the matrix pth root. *Numerical Algorithms*, 39(4):349–378, 2005. 4

[6] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6228–6237, 2018. 1

[7] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 2, pages 60–65. Ieee, 2005. 2

[8] Harold C Burger, Christian J Schuler, and Stefan Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *2012 IEEE conference on computer vision and pattern recognition*, pages 2392–2399. IEEE, 2012. 2

[9] Haoming Cai, Jingwen He, Yu Qiao, and Chao Dong. Toward interactive modulation for photo-realistic image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 294–303, 2021. 1, 3

[10] Yuanhao Cai, Xiaowan Hu, Haoqian Wang, Yulun Zhang, Hanspeter Pfister, and Donglai Wei. Learning to generate realistic noisy images via pixel-level noise-aware adversarial training. *Advances in Neural Information Processing Systems*, 34:3259–3270, 2021. 2

[11] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. *arXiv preprint arXiv:2204.04676*, 2022. 2, 3, 6, 7

[12] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018. 1

[13] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8):2080–2095, 2007. 2

[14] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image restoration by sparse 3d transform-domain collaborative filtering. In *Image Processing: Algorithms and Systems VI*, volume 6812, pages 62–73. SPIE, 2008. 2

[15] Weisheng Dong, Xin Li, Lei Zhang, and Guangming Shi. Sparsity-based image denoising via dictionary learning and structural clustering. In *CVPR 2011*, pages 457–464. IEEE, 2011. 2

[16] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006. 2

[17] Qingnan Fan, Dongdong Chen, Lu Yuan, Gang Hua, Nenghai Yu, and Baoquan Chen. Decouple learning for parameterized image operators. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 442–458, 2018. 1

[18] Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images. *IEEE transactions on image processing*, 16(5):1395–1411, 2007. 2

[19] Rich Franzen. Kodak lossless true color image suite. *source: http://r0k. us/graphics/kodak*, 4(2), 1999. 6

[20] Pascal Getreuer, Ignacio Garcia-Dorado, John Isidoro, Sungjoon Choi, Frank Ong, and Peyman Milanfar. Blade: Filter learning for general purpose computational photography. In *2018 IEEE International Conference on Computational Photography (ICCP)*, pages 1–11. IEEE, 2018. 2

[21] Shuhang Gu, Yawei Li, Luc Van Gool, and Radu Timofte. Self-guided network for fast image denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2511–2520, 2019. 2

[22] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. Weighted nuclear norm minimization with application to image denoising. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2862–2869, 2014. 2

[23] Shi Guo, Zifei Yan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. Toward convolutional blind denoising of real photographs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1712–1722, 2019. 3

[24] Jingwen He, Chao Dong, and Yu Qiao. Modulating image restoration with continual levels via adaptive feature modification layers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11056–11064, 2019. 1, 2, 3, 6

[25] Jingwen He, Chao Dong, and Yu Qiao. Interactive multi-dimension modulation with dynamic controllable residual learning for image restoration. In *European Conference on Computer Vision*, pages 53–68. Springer, 2020. 1, 3, 6

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[27] Nicholas J Higham. Newton's method for the matrix square root. *Mathematics of computation*, 46(174):537–549, 1986. 4

[28] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015. 6

[29] Lei Huang, Dawei Yang, Bo Lang, and Jia Deng. Decorrelated batch normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 791–800, 2018. 4

[30] Lei Huang, Yi Zhou, Fan Zhu, Li Liu, and Ling Shao. Iterative normalization: Beyond standardization towards efficient whitening. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4874–4883, 2019. 4, 5

[31] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[32] Jingyun Liang, Jiezhang Cao, Yuchen Fan, Kai Zhang, Rakesh Ranjan, Yawei Li, Radu Timofte, and Luc Van Gool. Vrt: A video restoration transformer. *arXiv preprint arXiv:2201.12288*, 2022. 2, 3, 7, 8

[33] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*, 2018. 2

[34] Kede Ma, Zhengfang Duanmu, Qingbo Wu, Zhou Wang, Hongwei Yong, Hongliang Li, and Lei Zhang. Waterloo exploration database: New challenges for image quality assessment models. *IEEE Transactions on Image Processing*, 26(2):1004–1016, 2016. 6

[35] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. *Advances in neural information processing systems*, 29, 2016. 3

[36] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001. 6

[37] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001. 6

[38] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11264–11272, 2019. 2

[39] Chong Mou, Yanze Wu, Xintao Wang, Chao Dong, Jian Zhang, and Ying Shan. Metric learning based interactive modulation for real-world super-resolution. *arXiv preprint arXiv:2205.05065*, 2022. 1, 3

[40] Yali Peng, Lu Zhang, Shigang Liu, Xiaojun Wu, Yu Zhang, and Xili Wang. Dilated residual networks with symmetric skip connection for image denoising. *Neurocomputing*, 345:67–76, 2019. 6

[41] Javier Portilla, Vasily Strela, Martin J Wainwright, and Eero P Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Transactions on Image processing*, 12(11):1338–1351, 2003. 2

[42] Wenqi Shao, Yixiao Ge, Zhaoyang Zhang, Xuyuan Xu, Xiaogang Wang, Ying Shan, and Ping Luo. Dynamic token normalization improves vision transformer. *arXiv preprint arXiv:2112.02624*, 2021. 2

[43] Wenqi Shao, Hang Yu, Zhaoyang Zhang, Hang Xu, Zhenguo Li, and Ping Luo. Bwcp: Probabilistic learning-to-prune channels for convnets via batch whitening. *arXiv preprint arXiv:2105.06423*, 2021. 4

[44] Dev Yashpal Sheth, Sreyas Mohan, Joshua L 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*, pages 1759–1768, 2021. 7, 8

[45] Chaofan Tao, Rui Lin, Quan Chen, Zhaoyang Zhang, Ping Luo, and Ngai Wong. Fat: Learning low-bitwidth parametric representation via frequency-aware transformation. *arXiv preprint arXiv:2102.07444*, 2021. 2

[46] 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. 2, 3, 7

[47] Chunwei Tian, Yong Xu, and Wangmeng Zuo. Image denoising using deep cnn with batch renormalization. *Neural Networks*, 121:461–473, 2020. 2

[48] 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*, pages 2157–2166, 2021. 8

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1

[50] Wei Wang, Ruiming Guo, Yapeng Tian, and Wenming Yang. Cfsnet: Toward a controllable feature space for image restoration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4140–4149, 2019. 1, 2, 3, 6

[51] Xintao Wang, Ke Yu, Chao Dong, Xiaoou Tang, and Chen Change Loy. Deep network interpolation for continuous imagery effect transition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1692–1701, 2019. 1, 3

[52] Jinjun Xu and Stanley Osher. Iterative regularization and nonlinear inverse scale space applied to wavelet-based denoising. *IEEE Transactions on Image Processing*, 16(2):534–544, 2007. 2

[53] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014. 6

[54] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5728–5739, 2022. 2, 6

[55] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017. 1, 2, 3, 6

[56] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3929–3938, 2017. 6

[57] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising.

IEEE Transactions on Image Processing, 27(9):4608–4622, 2018. 1, 2, 6

[58] Lei Zhang, Xiaolin Wu, Antoni Buades, and Xin Li. Color demosaicking by local directional interpolation and nonlocal adaptive thresholding. *Journal of Electronic imaging*, 20(2):023016, 2011. 6

[59] Zhaoyang Zhang, Yitong Jiang, Jun Jiang, Xiaogang Wang, Ping Luo, and Jinwei Gu. Star: A structure-aware lightweight transformer for real-time image enhancement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4106–4115, 2021. 1, 2

[60] Zhaoyang Zhang, Jingyu Li, Wenqi Shao, Zhanglin Peng, Ruimao Zhang, Xiaogang Wang, and Ping Luo. Differentiable learning-to-group channels via groupable convolutional neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3542–3551, 2019. 2

[61] Zhaoyang Zhang, Wenqi Shao, Jinwei Gu, Xiaogang Wang, and Ping Luo. Differentiable dynamic quantization with mixed precision and adaptive resolution. In *International Conference on Machine Learning*, pages 12546–12556. PMLR, 2021. 2