Supplemental Material for Robust Image Denoising using Kernel Predicting Networks

Zhilin Cai[2] Yang Zhang[1] Marco Manzi[1] Cengiz Oztireli[1] Markus Gross[1,2] Tunç Ozan Aydın[1]

DisneyResearch|Studios[1] ETH Zurich[2]

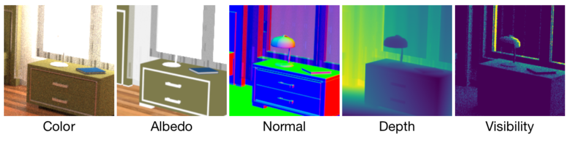

Figure 1: *Example KPCNs features used for denoising Monte Carlo renderings. Referenced in the main publication.*

	PSNR	SSIM	Training	Inference
full	38.60	0.948	3d 1h	74.8s
half bandwidths	38.14	0.939	2d 10h	43.2s
$\frac{5}{7}$ depth	38.19	0.940	2d 13h	50.6s
single scale	37.64	0.931	1d 21h	33.9s

Table 1: *PSNR/SSIM results and training/inference times of various size-reduced KPCN generalizers.*

1. Ablation Studies

In this supplemental material we present various ablations of our denoising method.

1.1. Inference Time vs. Denoising Quality

We investigated how various modifications to the KPCN generalizer with the goal of reducing inference and training times affect the denoising quality through an ablation study on the SIDD dataset. Table 1 shows a comparison between the full Robust KPCN that we described in Section 4 and several other configurations where we halve the bandwidths of each layer of the KPCN generalizer, reduce the depth of the KPCN generalizer. Note that all configurations still use 5 specialized U-Net denoisers, which are factored into the reported time measurements. The results show each of these approaches could be viable ways to reduce the inference and training times with modest sacrifices in denoising quality.

In applications when the noise magnitude of test images are known to be relatively low, another strategy could be to reduce the capacity of specialized denoisers.

To this end we produced 9200 training pairs of clean and noisy images, where the clean images were obtained from the publicly available dataset [ZSS*18], and the noisy images were produced by adding Gaussian Noise at a magnitudes $\sigma = \{15,25,35\}$ to the corresponding clean images. We also trained a KPCN generalizer (as discussed in Section 3) by utilizing the 3 specialized DnCNN denoisers (denoted as DnCNN-S) trained for $\sigma = \{15,25,35\}$ to produce denoised-image features. Our testing dataset was similarly generated using 264 different images from the same dataset, to which we added Gaussian noise at magnitudes sampled from the range $\sigma \in [5,25]$. Table 2 shows a comparison on the aforementioned dataset, between the full Robust KPCN we described also in Section 3, and another variation where we reduce the capacity of the specialized DnCNN denoisers to a third of the originals. The capacity reduction results in minor reduction in denoising quality, and as such could be employed as another viable strategy while optimizing overall efficiency.

Noise level	$\sigma = 5$	$\sigma = 10$	$\sigma = 15$	$\sigma = 20$	$\sigma = 25$
full	40.18	39.23	38.94	38.59	38.37
$\frac{1}{3}$ DnCNN-S	39.93	38.90	38.56	38.02	37.85

Table 2: *PSNR results of Robust KPCN with denoised-image features produced by a full DnCNN-S vs. a computationally more efficient size-reduced DnCNN-S.*

1.2. Denoised-image Feature Selection

The quality and quantity of denoised-image features can have a significant impact on Robust KPCN's denoising quality. In Table 3 we compare a baseline KPCN that utilizes 3 DnCNN-S denoisers (trained for noise magnitudes $\sigma = \{10,15,20\}$) with various other denoised-image feature configurations on the dataset from Section 1.1. Note that in this experiment we train each of these Robust KPCN configurations from scratch. The first configuration which does not use any denoised-image features results in a notable reduction in denoising quality with respect to the baseline configuration beyond relatively low noise magnitudes. The next configuration shows the results of using the outcome of BM3D as the sole denoised-image feature in our Robust KPCN framework. In our experiments we observed that the denoising quality of Robust KPCN depends highly on the quality of individual denoised-image

Noise level	$\sigma = 5$	$\sigma = 10$	$\sigma = 15$	$\sigma = 20$	$\sigma = 25$
no feats	39.07	38.31	38.04	36.62	35.93
BM3D feat	38.93	37.94	37.71	36.22	35.11
3 DnCNN-S feats	40.18	39.23	38.94	38.59	38.37
5 DnCNN-S feats	40.46	39.48	38.92	38.60	38.41

Table 3: *PSNR results of Robust KPCN with various features on different Gaussian noise levels.*

features. In fact in this experiment using BM3D denoised images as a features results in worse results than using no denoised-image features at all. This outcome suggests against the potential strategy of using additional low-quality but computationally cheap features for improving overall denoising quality.

We also investigate another Robust KPCN configuration where we use 5 specialized denoisers (trained for noise magnitudes $\sigma = \{5, 10, 15, 20, 25\}$) as opposed to 3 specialized denoisers of the base configuration. This modification results only in minor improvements in denoising quality, which suggests that increasing the number of features beyond a certain point results in diminishing returns.

1.3. Loss Function Selection

We tested various loss functions by using them to optimize our KPCN generalizer by training on the SIDD medium dataset, and evaluated their individual performances on the corresponding validation set. Figure 2 shows convergence plots of KPCN generalizers optimized with MAPE, L_1, MRSE, and SSIM, e.g. (a) shows MAPE, L_1, MRSE, and SSIM errors on the validation set while training the KPCN generalizer with MAPE. Overall training with MAPE results in the lowest loss across all error metrics we tested for, and shows relatively little fluctuations during training. Hence throughout this paper we used MAPE to optimize all our KPCN generalizers.

We also note that, despite corresponding better to subjective similarity, SSIM is often difficult to optimize for. Hence using SSIM as the KPCN generalizer loss function results in overfitting. A possible venue for future exploration could be to pre-train a KPCN generalizer using MAPE, and later to switch to SSIM loss for fine tuning.

References

[ZSS*18] Zamir A. R., Sax A., Shen W. B., Guibas L. J., Malik J., Savarese S.: Taskonomy: Disentangling task transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018). 1

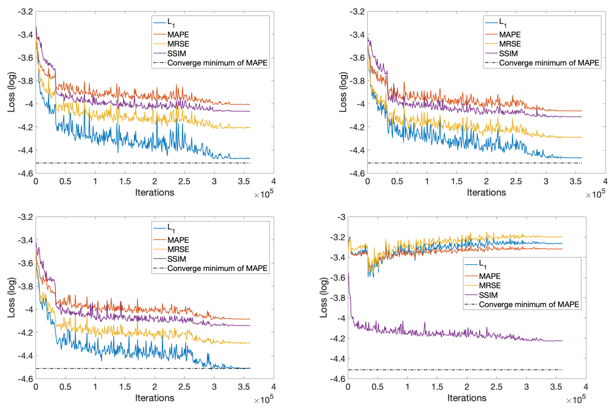

Figure 2: *Convergence plots of KPCN generalizers optimized with MAPE, L_1, MRSE, and SSIM (shown in that order left to right.)*