

FAST, TRAINABLE, MULTISCALE DENOISING

Sungjoon Choi, John Isidoro, Pascal Getreuer, Peyman Milanfar

Google Research

{sungjoonc, isidoro, getreuer, milanfar}@google.com

ABSTRACT

Denoising is a fundamental imaging problem. Versatile but fast filtering has been demanded for mobile camera systems. We present an approach to multiscale filtering which allows real-time applications on low-powered devices. The key idea is to learn a set of kernels that upscales, filters, and blends patches of different scales guided by local structure analysis. This approach is trainable so that learned filters are capable of treating diverse noise patterns and artifacts. Experimental results show that the presented approach produces comparable results to state-of-the-art algorithms while processing time is orders of magnitude faster.

Index Terms— Image denoising, filter learning, multiscale

1. INTRODUCTION

Image denoising is known as a challenging problem that has been explored for many decades. Patch matching methods [1, 2, 3, 4, 5] exploit repetitive textures and produce high quality results thanks to more accurate weight computation than bilateral filtering [6, 7]. However, higher computational complexity limits their application for low-powered devices. Recently, deep learning based approaches [8, 9] have become popular. While trainable networks achieve generic and flexible processing capabilities, deep layers are hard to analyze and are even more computationally expensive than patch based methods, making them harder to use in real-time applications.

Meanwhile, multiscale strategies have been widely adopted for various problems in the signal processing and computer vision communities [10, 11, 12]. Multiscale techniques effectively increase the footprint of filter kernels while introducing minimal overhead and allow for more efficient application of filtering than fixed-scale kernels. Consequently, it is natural to take advantage of the multiscale approach for denoising.

Our work makes two contributions. First, we introduce a “shallow” learning framework that trains and filters very fast using local structure tensor analysis on color pixels. Because it has only a few convolution layers, the set of resulting filters is easy to visualize and analyze. Second, we cascade the learning stage into a multi-level pipeline to effectively filter larger areas with small kernels. In each stage of the pipeline,

we train filtering that jointly upscales coarser level $(\ell + 1)$ and denoises and blends finer level ℓ .

2. RELATED WORK

The influential non-local means (NLM) filtering [13] has received great interest since its introduction. NLM generalizes bilateral filtering by using patch-wise photometric distance to better characterize self-similarity, but at increased computational cost. Many techniques have been proposed to accelerate NLM [1, 14]. [15] uses a multiscale approach to perform NLM filtering at each level of a Laplacian pyramid. The pull-push NLM [16] method constructs up and down pyramids where NLM weights are fused separately.

Sparsity methods open a new chapter in denoising. The now classic block-matching and 3D filtering (BM3D) [3] based on 3D collaborative Wiener filtering is considered to be state-of-the-art for Gaussian noise. Nonlocally centralized sparse representation (NCSR) [17] introduces a sparse model that can be solved by a conventional iterative shrinkage algorithm.

Learning-based methods have also become popular in image processing recently. Trainable nonlinear reaction diffusion (TNRD) [18] uses multi-stage trainable nonlinear reaction diffusion as an alternative to CNNs where the weights and the nonlinearity is trainable. Rapid and accurate image super resolution (RAISR) [19] is an efficient edge-adaptive image upscaling method that uses structure tensor features to select a filter at each pixel from among a set of trained filters. Best linear adaptive enhancement (BLADE) [20] generalizes RAISR to a two-stage shallow framework applicable to a diverse range of imaging tasks, including denoising.

3. FILTER LEARNING

We begin with BLADE filter learning. The framework in BLADE [20] is formulated for filtering an image at a single scale. We extend BLADE to a trainable multi-level filter framework for denoising, using noisy and noise-free images as training pairs.

Spatially-adaptive filtering. The input image is denoted by \mathbf{z} and the value at pixel location $i \in \Omega \subset \mathbb{Z}^2$ is denoted by

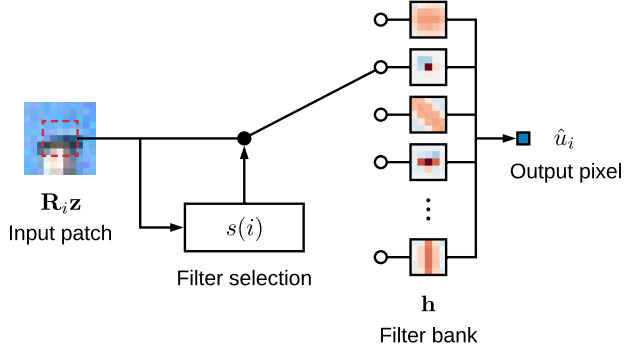


Fig. 1. Two stage spatially-adaptive filtering. For a given output pixel \hat{u}_i , we only need to evaluate the one linear filter that is selected by $s(i)$.

z_i . Spatially-adaptive filtering operates with a set of linear FIR filters $\mathbf{h}^1, \dots, \mathbf{h}^K$. h_j^k denotes a filter value of \mathbf{h}^k , where $j \in F \subset \mathbb{Z}^2$ and F is the footprint of the filter. The main idea of BLADE is that a different filter is selected by a function $s : \Omega \rightarrow \{1, \dots, K\}$ for each output pixel,

$$\hat{u}_i = \sum_{j \in F} h_j^{s(i)} z_{i+j}. \quad (1)$$

Or in vector notation, the i th output pixel is

$$\hat{u}_i = (\mathbf{h}^{s(i)})^T \mathbf{R}_i \mathbf{z}, \quad (2)$$

where \mathbf{R}_i is an operator that extracts a patch centered at i . Fig. 1 depicts the two stage pipeline that adaptively selects one filter from a linear filterbank for each pixel.

Filter selection. Filter selection should segregate input patches so that the relationship to the corresponding target pixels is well-approximated by a linear estimator, while keeping a manageable number of filters. Filter selection should also be robust to noise, and computationally efficient. In this light, we use features of the structure tensor. While [19, 20] use the structure tensor of the image luma channel, we find that analysis of luma alone occasionally misses key structures that are visible in color, as shown in Fig. 2. We find it beneficial for denoising to compute a structure tensor jointly using all color channels, as suggested previously for instance by Weickert [21]. Structure tensor analysis provides a robust local gradient estimate by principal components analysis (PCA) of the gradients over pixel i 's neighborhood N_i , as the minimizer of

$$\arg \min_{\mathbf{a}} \sum_{j \in N_i} w_j^i (\mathbf{a}^T \mathbf{g}_j)^2 \quad (3)$$

where \mathbf{g}_j is the gradient at pixel location j and w_j^i is a spatial weighting. With a 2×3 Jacobian matrix for color-wise gradients

$$\mathbf{G}_j = [\mathbf{g}_j^R \quad \mathbf{g}_j^G \quad \mathbf{g}_j^B], \quad (4)$$

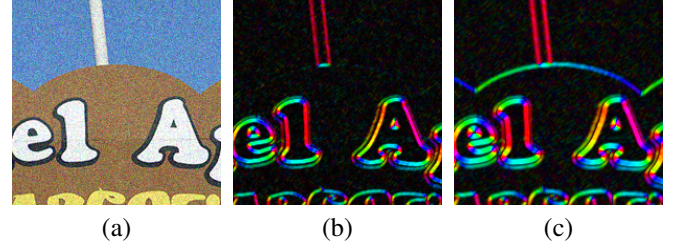


Fig. 2. Visualization of structure analysis where estimated orientations and strengths are mapped to hue and value, respectively. (a) Input image. (b) Structure analysis of [19]. (c) Our structure analysis. Note that strong edges are not detected in (b) which results in a blurry reconstruction.

we now find a unit vector \mathbf{a} minimizing

$$\sum_j w_j^i \|\mathbf{a}^T \mathbf{G}_j\|^2 = \mathbf{a}^T \left(\sum_j w_j^i \mathbf{G}_j \mathbf{G}_j^T \right) \mathbf{a} = \mathbf{a}^T \mathbf{T}_i \mathbf{a}. \quad (5)$$

The spatially-filtered structure tensor \mathbf{T}_i is

$$\mathbf{T}_i = \sum_c \sum_j w_j^i \begin{bmatrix} g_{x,j}^c g_{x,j}^c & g_{x,j}^c g_{y,j}^c \\ g_{x,j}^c g_{y,j}^c & g_{y,j}^c g_{y,j}^c \end{bmatrix} \quad (6)$$

where $c \in \{R, G, B\}$ and $(g_{x,j}^c, g_{y,j}^c)^T = \mathbf{g}_j^c$. For each pixel i , eigenanalysis of the 2×2 matrix \mathbf{T}_i explains the variation in the gradients along the principal directions. The unit vector \mathbf{a} minimizing $\mathbf{a}^T \mathbf{T}_i \mathbf{a}$ is the eigenvector of \mathbf{T}_i corresponding to the smallest eigenvalue, which forms the orientation feature. The square root of the larger eigenvalue λ_1 is a smoothed estimate of the gradient magnitude [22]. In addition, we use coherence $(\sqrt{\lambda_1} - \sqrt{\lambda_2}) / (\sqrt{\lambda_1} + \sqrt{\lambda_2})$ from the eigenvalues $\lambda_1 \geq \lambda_2$, which ranges from 0 to 1 and characterizes the degree of local anisotropy. We use these three features for filter selection s to index a filter in the filterbank.

Given a target image \mathbf{u} and its pixel value u_i at pixel location i , we formulate filter learning as

$$\arg \min_{\mathbf{h}^1, \dots, \mathbf{h}^K} \|\mathbf{u} - \hat{\mathbf{u}}\|^2 \quad (7)$$

$$\begin{aligned} \|\mathbf{u} - \hat{\mathbf{u}}\|^2 &= \sum_{k=1}^K \sum_{\substack{i \in \Omega: \\ s(i)=k}} |u_i - \hat{u}_i|^2 \\ &= \sum_{k=1}^K \sum_{\substack{i \in \Omega: \\ s(i)=k}} |u_i - (\mathbf{h}^k)^T \mathbf{R}_i \mathbf{z}|^2 \end{aligned} \quad (8)$$

which amounts to a multivariate linear regression for each filter \mathbf{h}^k , described in detail in [20]. The above training and filtering steps are repeated for each color channel¹.

¹To denoise color images, images are converted to YCbCr (ITU-R BT.601). We train filters separately on Y, Cb, and Cr while using the same filter selector $s(\cdot)$ to capture channel-specific noise statistics.

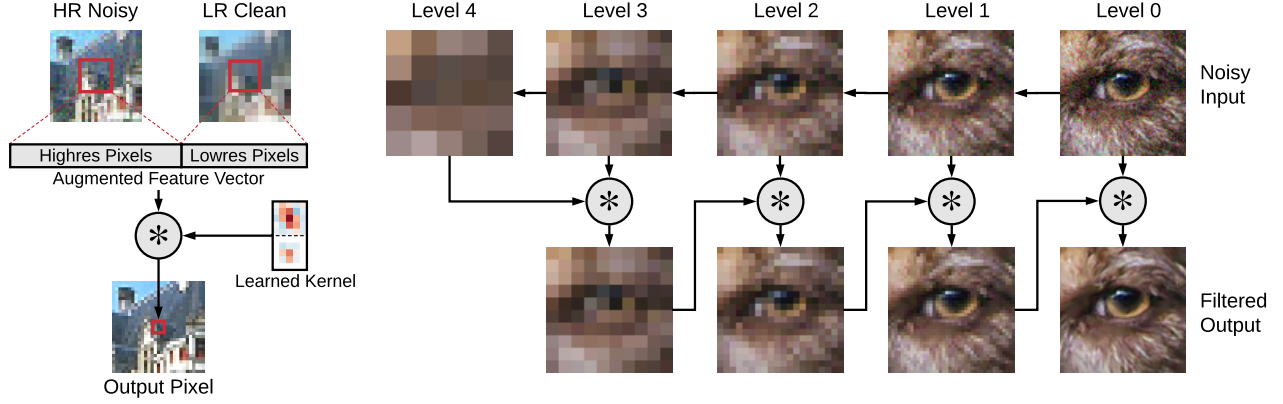


Fig. 3. Overview of multiscale denoising. (left) A filter kernel is learned so that it upscales a coarser-level filtered image, filters a finer-level noisy image, and blends them into a target pixel. (right) Combined together, cascaded learned filters form a large and irregular kernel and effectively remove noise on variable structure.

4. MULTISCALE DENOISING

In this section, we describe multiscale denoising. The overview of the pipeline is described in Fig. 3.

Fixed-scale filtering. The framework described in Section 3 can be trained from pairs of noisy and clean images to perform denoising. Based on the noise in the training data, denoisers for different kinds of noise can be trained. For example, [20] shows that BLADE can perform both AWGN denoising and JPEG compression artifact removal, interpreting JPEG artifacts as noise. Other more complex noise models or real world noise could be learned thanks to the generic trainable framework.

Fig. 4 visualizes learned filters for AWGN noise where $\sigma = 20$. Fig. 5 shows the denoised results with the filters trained for different noise levels. Fixed-scale filtering is effective for the low noise level while it exhibits insufficient power for stronger noise because the footprint of the used filter (7×7) is too small to compensate the noise variance. Increasing the size of the filters is undesirable as it increases the time complexity quadratically.

Multiscale filtering. We consider the fixed-scale denoising filter as a building block for multiscale training and inference. We begin by taking noisy input images and forming pyramids by downsampling by factors of two. Standard bicubic downsampling is enough to effectively reduce noise level by half, and is extremely fast. Pyramids of target images are constructed in the same fashion.

We start training from the second from the coarsest level L . To compute the output $\hat{\mathbf{u}}^\ell$ at level ℓ , the filters \mathbf{f}^ℓ upscale the next coarser output $\hat{\mathbf{u}}^{\ell+1}$ and filters \mathbf{g}^ℓ denoise and blend with the current level’s noisy input \mathbf{z}^ℓ ,

$$\hat{u}_i^\ell = \sum_{j \in F} f_j^{\ell, s(i)} \hat{u}_{i/2+j}^{\ell+1} + \sum_{j \in F} g_j^{\ell, s(i)} z_{i+j}^\ell, \quad (9)$$

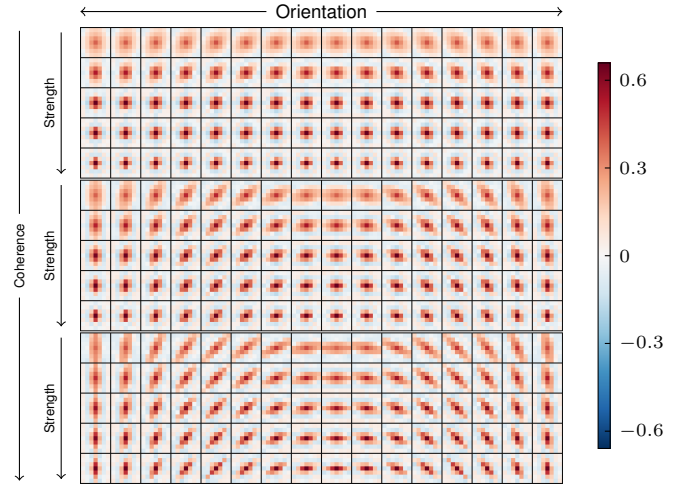


Fig. 4. 7×7 filters for AWGN denoising with noise standard deviation 20, 16 different orientations, 5 strength values, and 3 coherence values.



Fig. 5. Results of fixed-scale denoising. (left) Low noise input ($\sigma = 15$). (right) High noise input ($\sigma = 50$).

or denoting the filter pair by $\mathbf{h}^{\ell, k} = [\mathbf{f}_{\ell, k}^\ell, \mathbf{g}_{\ell, k}^\ell]^\top$, as

$$\hat{u}_i^\ell = (\mathbf{h}^{\ell, s(i)})^\top \begin{bmatrix} \mathbf{R}_{i/2} \hat{\mathbf{u}}^{\ell+1} \\ \mathbf{R}_i \mathbf{z}^\ell \end{bmatrix} \quad (10)$$

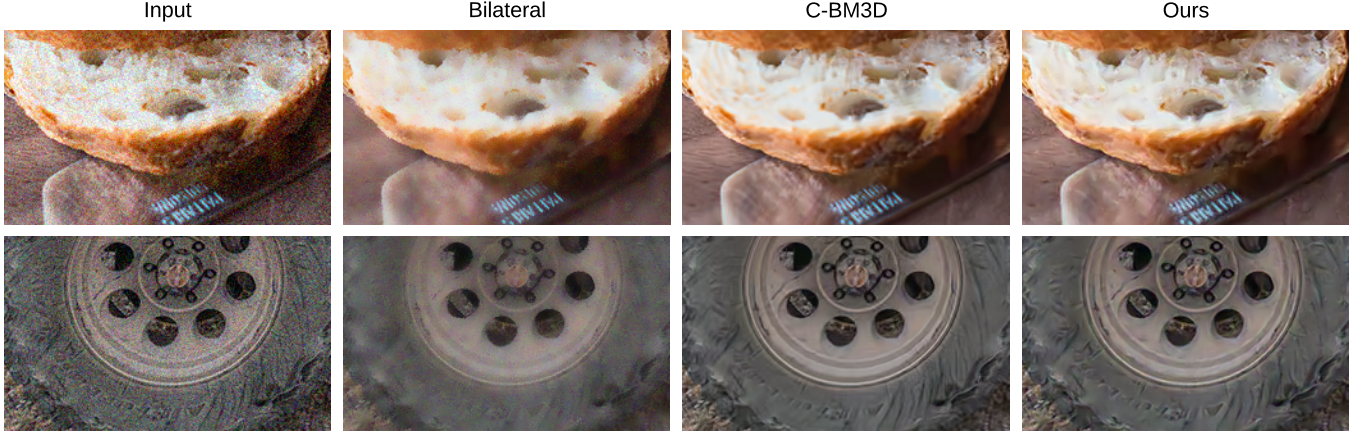


Fig. 6. Qualitative comparisons with noisy input $\sigma = 20$.

where the base case $\hat{\mathbf{u}}^L = \mathbf{z}^L$ as illustrated in Fig. 3 (a). We train filters by using input patches from the current level ℓ and corresponding patches from the filtered output at the next coarser level $\ell + 1$. Once level ℓ is trained, filtered images $\hat{\mathbf{u}}^\ell$ are computed and then consumed for training $\mathbf{h}^{\ell-1}$ at the next finer level.

Overall, our shallow inference can be performed with high efficiency. Both the selection s and filtering are vectorization and parallelization-friendly because most operations are additions and multiplications on sequential data with few dependencies. On the Pixel 2017 phone, inference time is 18 MP/s on CPU and 188 MP/s on GPU.

5. EXPERIMENTAL RESULTS

We have evaluated the presented pipeline on 68 test images of the Berkeley dataset [23]. We used high quality images separately collected from the Internet to train a filterbank where about 2.23×10^9 pixels were consumed. Noisy images were synthesized with an AWGN model, and then quantized and clamped at 8-bit resolution. To get more samples, we included spatial axial flips and 90° rotations of each observation and target image pair in the training set so that the filters learn symmetries. At every level, we used 16 orientation buckets, 16 strength buckets, and 16 coherence buckets for structure analysis. For low noise level $\sigma < 10$, the filter size of 5×5 was used for finer level and 3×3 for coarser level. Otherwise, the filter size of 7×7 was used for finer level and 5×5 for coarser level. The level of pyramid is set so that the noise standard deviation of the coarsest level is less than 2.

Table 1 reports the PSNR and timing of various state-of-the-art techniques. We provided each method with the same noise variance parameter used to synthesize noisy input. Running times were measured on a workstation with an Intel Xeon E5-1650 3.5 GHz CPU. The results of the proposed pipeline are comparable to the state-of-the-art algorithms as

Table 1. Quantitative evaluation with the Berkeley dataset.

| Method | PSNR (dB) | | | Time (s) |
|---------------------------|---------------|---------------|---------------|----------|
| | $\sigma = 15$ | $\sigma = 25$ | $\sigma = 50$ | |
| BM3D [3] | 30.87 | 28.20 | 24.63 | 47.2 |
| K-SVD [2] | 30.66 | 27.82 | 23.80 | 35.8 |
| TNRD $_{7 \times 7}$ [18] | 31.18 | 28.48 | 24.75 | 16.5 |
| C-BM3D [24] | 33.24 | 30.18 | 25.85 | 21.9 |
| Ours | 32.46 | 29.58 | 25.92 | 0.038 |

For methods shaded with gray, color channels were jointly denoised; otherwise the filters were independently applied on each channel. Running times were measured on 1 MP images.

shown in Fig. 6 while it is orders of magnitude faster. Per-image processing time of ours was linear to the number of pixels in the image.

6. CONCLUSION

We have presented a trainable multiscale approach for denoising. The key idea is to learn filters that jointly upscale, blend, and denoise between successive levels. The learning process is capable of treating diverse noise patterns and artifacts. Experiments demonstrate that the presented approach produces results comparable to state-of-the-art algorithms with processing time that is orders of magnitude faster.

The presented pipeline is not perfect. For inference, we assumed the noise level of input image is known and used the filters trained with the data of the same noise level. There are many ways to estimate the level of noises, which can guide us to select the right filter set. Also we assumed the noise level is uniform across pixel locations. We believe we can characterize and model the noise response of a camera system, and then integrate this information into filter selection.

7. REFERENCES

- [1] A. Buades, B. Coll, and J.-M. Morel, “A review of image denoising algorithms, with a new one,” *Multi-scale Modeling & Simulation*, vol. 4, no. 2, pp. 490–530, 2005.
- [2] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” *IEEE Trans. on Image Processing*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006.
- [3] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, “Image denoising by sparse 3-D transform-domain collaborative filtering,” *IEEE Trans. on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.
- [4] C. Kervrann and J. Boulanger, “Local adaptivity to variable smoothness for exemplar-based image regularization and representation,” *International Journal of Computer Vision*, vol. 79, no. 1, pp. 45–69, 2008.
- [5] L. Zhang, W. Dong, D. Zhang, and G. Shi, “Two-stage image denoising by principal component analysis with local pixel grouping,” *Pattern Recognition*, vol. 43, pp. 1531–1549, Apr. 2010.
- [6] S. M. Smith and J. M. Brady, “SUSAN – a new approach to low level image processing,” *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, 1997.
- [7] C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images,” in *International Conference on Computer Vision*, January 1998, pp. 836–846.
- [8] J. Xie, L. Xu, and E. Chen, “Image denoising and inpainting with deep neural networks,” in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., pp. 341–349. 2012.
- [9] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising,” *IEEE Trans. on Image Processing*, vol. 26, no. 7, pp. 3142–3155, July 2017.
- [10] C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden, “Pyramid methods in image processing,” 1984.
- [11] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, “The lumigraph,” in *ACM SIGGRAPH*, 1996, pp. 43–54.
- [12] S. Paris, S. Hasinoff, and J. Kautz, “Local Laplacian filters: Edge-aware image processing with a Laplacian pyramid,” *ACM Trans. Graph.*, vol. 30, no. 4, pp. 68–1, 2011.
- [13] A. Buades, B. Coll, and J.-M. Morel, “A non-local algorithm for image denoising,” in *Conference on Computer Vision and Pattern Recognition*, 2005, vol. 2, pp. 60–65.
- [14] X. Liu, X. Feng, and Y. Han, “Multiscale nonlocal means for image denoising,” in *International Conference on Wavelet Analysis and Pattern Recognition*, 2013.
- [15] S. Nercessian, K. A. Panetta, and S. S. Agaian, “A multiscale non-local means algorithm for image de-noising,” *SPIE*, vol. 8406, pp. 84060J–84060J–10, 2012.
- [16] J. Isidoro and P. Milanfar, “A pull-push method for fast non-local means filtering,” in *International Conference on Image Processing*, 2016, pp. 1968–1972.
- [17] W. Dong, L. Zhang, G. Shi, and X. Li, “Nonlocally centralized sparse representation for image restoration,” *IEEE Trans. on Image Processing*, vol. 22, no. 4, pp. 1620–1630, 2013.
- [18] Y. Chen and T. Pock, “Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1256–1272, 2017.
- [19] Y. Romano, J. Isidoro, and P. Milanfar, “RAISR: Rapid and Accurate Image Super Resolution,” *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 110–125, 2017.
- [20] P. Getreuer, I. Garcia-Dorado, J. Isidoro, S. Choi, F. Ong, and P. Milanfar, “BLADE: Filter Learning for General Purpose Computational Photography,” *ArXiv e-prints*, Nov. 2017.
- [21] J. Weickert, “Coherence-enhancing diffusion of colour images,” in *National Symposium on Pattern Recognition and Image Analysis*, 1997, vol. 1, pp. 239–244.
- [22] X. Feng and P. Milanfar, “Multiscale principal components analysis for image local orientation estimation,” in *Asilomar Conference on Signals, Systems and Computers*, November 2002, vol. 1, pp. 478–482.
- [23] 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 *International Conference on Computer Vision*, July 2001, vol. 2, pp. 416–423.
- [24] K. Dabov, A. Foi, V. Katkovnik, and K. O. Egiazarian, “Color image denoising via sparse 3d collaborative filtering with grouping constraint in luminance-chrominance space,” in *International Conference on Image Processing*, 2007, pp. 313–316.