Basis Prediction Networks for Effective Burst Denoising with Large Kernels

Zhihao Xia[1], Federico Perazzi[2], Michaël Gharbi[2], Kalyan Sunkavalli[2], Ayan Chakrabarti[1]
[1]Washington University in St. Louis [2]Adobe Research
{zhihao.xia,ayan}@wustl.edu, {perazzi,mgharbi,sunkaval}@adobe.com

Abstract

Bursts of images exhibit significant self-similarity across both time and space. This motivates a representation of the kernels as linear combinations of a small set of basis elements. To this end, we introduce a novel basis prediction network that, given an input burst, predicts a set of global basis kernels — shared within the image — and the corresponding mixing coefficients — which are specific to individual pixels. Compared to state-of-the-art techniques that output a large tensor of per-pixel spatiotemporal kernels, our formulation substantially reduces the dimensionality of the network output. This allows us to effectively exploit comparatively larger denoising kernels, achieving both significant quality improvements (over 1dB PSNR) and faster run-times over state-of-the-art methods.

1. Introduction

Burst denoising algorithms [24, 12, 27] seek to enable high-quality photography in challenging conditions and are increasingly being deployed in commercial mobile cameras [12, 21]. A burst captures a sequence of short-exposure frames of the scene that are free of motion-blur, but with a high amount of noise in each frame and relative motion between frames. By accounting for this relative motion and using the fact that the noise is independent across frames, burst denoising attempts to aggregate these inputs and predict a single noise- and blur-free image estimate.

Recently, Mildenhall *et al*. [27] proposed an elegantly simple yet surprisingly successful approach to burst denoising. Rather than explicitly estimating inter-frame motion [12, 24, 13, 14, 18], their method produces denoised estimates at each pixel as a weighted average of observed noisy intensities in a window around that pixel's location in all frames. These averaging weights, or kernels, are allowed to vary from pixel-to-pixel to implicitly account for motion and image discontinuities, and are predicted from the noisy input burst using a "kernel prediction network" (KPN).

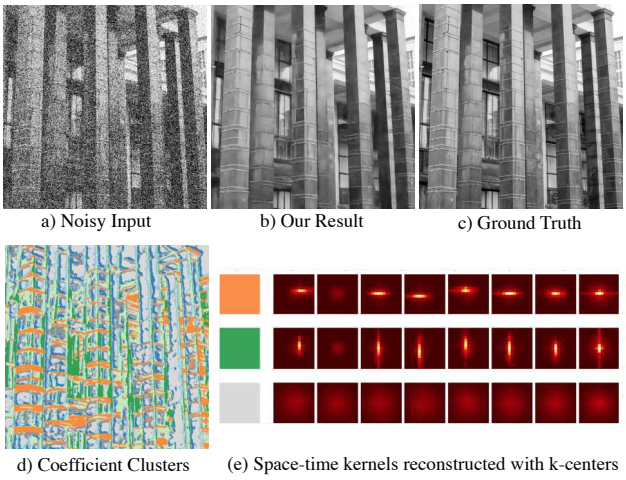

Figure 1: (*top*) Qualitative result of our denoising network. The proposed approach (b) recovers fine geometric details from a very noisy image burst (a, only one frame is shown). (*Bottom*) our per-burst bases can compactly represent a large variety of kernels by exploiting the redundancy of local image structures. We clustered the per-pixel coefficients predicted from the noisy burst using k-means. The clusters we obtain show strong spatial structure (d). For instance, we can identify a cluster corresponding to horizontal image edges (e, orange), one corresponding to vertical edges (e, green), and another for homogeneous regions with no structure (e, gray).

However, KPNs need to produce an output that is significantly higher-dimensional than the denoised image—even for 5×5 kernels with eight frames, a KPN must predict 400 times as many kernel weights as image intensities. This comes with significant memory and computational costs, as well as difficulty in training given the many degrees of freedom in the output. As a result, KPNs have so far been used only with small kernels. This limits their denoising ability by preventing averaging across bigger spatial regions, and over frames with larger relative motion.

In this paper, we introduce an approach to predict large

Part of this work was done when ZX was an intern at Adobe Research.

denoising kernels, and thus benefit from wider aggregation, while simultaneously limiting output dimensionality at each pixel, making the prediction network easier to train and compute- and memory-efficient. This approach is motivated by a long history of successful image restoration methods that have leveraged internal structure and self-similarity in natural images [5, 2, 35, 22, 40]. In burst denoising, self-similarity is particularly strong because we expect both *spatial* structure in the form of similar patterns that recur within a frame and across frames of the same scene, and *temporal* structure caused by consistency in scene and camera motion. Given the expected self-similarity and structure in the image intensities themselves, we argue that the corresponding denoising kernels must also have similar structure. Specifically, while allowing for individual per-pixel denoising kernels to be large, we assume that all the kernels for a given image span a lower-dimensional subspace.

Based on this observation, we introduce a new method for kernel-based burst denoising that achieves better accuracy and efficiency. Our contributions are:

- We introduce basis prediction networks (BPNs): given an input noisy burst, these networks predict a global low-dimensional basis set for large denoising kernels, and per-pixel coefficient vectors relative to this basis. BPN outputs are thus significantly lower-dimensional than the per-pixel arbitrary kernels from a regular KPN.
- Enforcing this structure on the denoising kernels acts as a form of regularization, and our experiments show this leads to state-of-the-art denoising performance with significantly higher quality (>1 dB PSNR) than KPNs.
- Beyond reducing memory usage and computational burden at the output layer, the structure of BPN's output enables the final kernel filtering step to be performed much more efficiently in the Fourier domain. We show this in terms of required number of FLOPs, as well as experimentally with actual run-times.

2. Related Work

Single-image and video denoising. Single-image denoising has been studied extensively. To overcome the ill-posed nature of the problem, classical approaches [32, 33, 42] developed regularization schemes that model the local statistics of natural images. The most successful approaches [5, 2] exploit non-local self-similarity within the image and denoise pixels by aggregating similar pixels or patches from distant regions. These methods have been extended to denoise videos [19, 25], where the search for similar patches proceeds not only within frames, but also across different frames. Recent work has improved image denoising performance using convolutional networks trained on large datasets [3, 38, 39, 22, 40, 35, 36]. Other works use hand-crafted features to select per-pixel filters from learned filter sets for efficient image enhancement [30, 8].

Burst denoising. Single-image denoising is a fundamentally under-constrained problem. Burst processing can reduce this ambiguity by using multiple observations (the frames of the burst) to recover a noise-free depiction of the scene. Burst denoising algorithms are now extensively used in commercial smartphone cameras [12] and can produce compelling results even in extreme low light scenarios [4, 21]. Like in video denoising, a significant challenge for burst processing is the robustness to inter-frame motion. Many methods explicitly estimate this motion to align and denoise the frames [12, 24, 13, 14, 18]. Current state-of-the-art burst denoising techniques [27, 18, 11, 26] are based on deep neural networks. Many of them only require coarse registration, relying on the network to account for the small residual misalignments [27, 11, 26].

Kernel Prediction Networks. Given a burst sequence, Mildenhall et al. [27] predict per-pixel kernels that are then applied to the input burst to produce the denoised output. They demonstrate that KPNs outperform direct pixel synthesis that produces oversmooth results. This idea has been extended to use multiple varying size kernels at every pixel [26]. KPNs have also been used in other applications, including denoising Monte Carlo renderings [1, 34, 9], video super-resolution [16] and deblurring [41], frame interpolation [28, 29, 23] and video prediction [15, 6, 23, 37].

Given their high-dimensional output (per-pixel kernels, that are three-dimensional in the case of burst denoising), KPNs have significant memory and compute requirements. Marinč*et al*. [26] and Niklaus *et al*. [29] ameliorate this by predicting spatially separable kernels: forming spatial kernels as outer products of predicted horizontal and vertical kernels. However, this makes a strong a-priori assumption about kernel structure, and still requires constructing and filtering with different per-pixel kernels. In contrast, our approach assumes that the set of per-pixel kernels for a scene span a low-dimensional sub-space, and predicts a basis for this sub-space based on the burst input. This approach also allows us to benefit from fast filtering in the Fourier domain.

3. Method

In burst denoising, we are given an input noisy burst of images $I[n, t]$, where n indexes spatial locations and $t \in \{1, \dots, T\}$ the different frames in the burst. Using a heteroscedastic Gaussian noise model [7], which accounts for both read and shot noise, we relate this to the corresponding noise-free frames $R[n, t]$ as:

$$I[n, t] \sim \mathcal{N}(R[n, t], \sigma_r^2 + \sigma_s^2 R[n, t]), \tag{1}$$

where σ_r^2 and σ_s^2 are the read- and shot-noise parameters. Choosing the first frame as reference, our goal is to produce a single denoised image $\hat{R}[n]$ as an estimate of the first noise-free frame $R[n, 1]$.

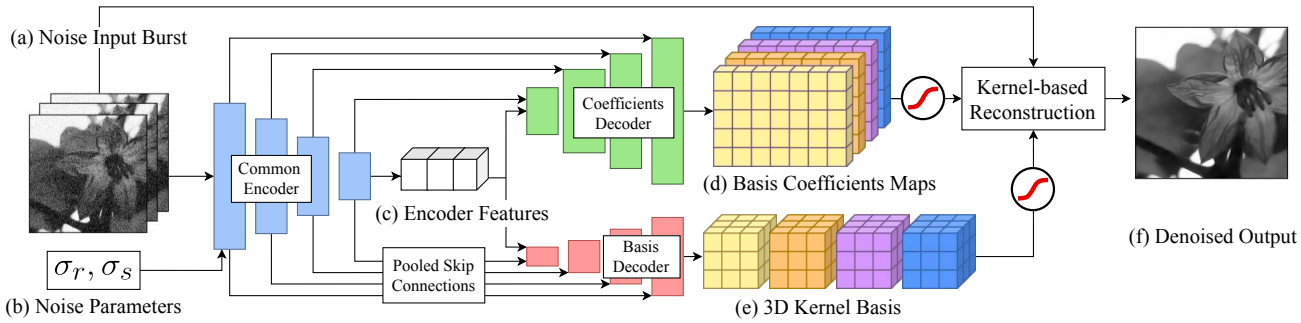

Figure 2: Our basis prediction network takes as input a burst of noisy input frames (a) together with the noise parameters (b). The frames are encoded into a shared feature space (c). These features are then decoded by two decoders with skip connections into a burst-specific basis of 3D kernels (e) and a set of per-pixel mixing coefficients (d). Both the coefficients and basis kernels are individually unit-normalized. Finally, we obtain per-pixel kernels by mixing the basis elements according to the coefficients and we apply them to the input burst to produce the final denoised image (f).

3.1. Kernel-based burst denoising

Rather than train a network to regress $\hat{R}$ directly, Kernel Prediction Networks output a field of denoising kernels $w_n[\delta, t]$, one for each pixel n at each frame t. The kernels have a spatial support $K \times K$, indexed by δ, with separate weights for each frame. Given these predicted kernels, the denoised estimate $\hat{R}$ is formed as:

$$\hat{R}[n] = \sum_t \sum_\delta w_n[\delta, t] I[n - \delta, t]. \tag{2}$$

A key bottleneck in this pipeline is the prediction of this dense kernel field w, which requires producing K^2T numbers at *every* pixel of the output. Since networks with high-dimensional outputs are both expensive and require learning a large number of parameters in their last layer, KPNs have typically been used only with small kernels ($K = 5$ in [27]).

3.2. Basis Prediction Networks

Instead of directly predicting unconstrained kernels for each spatial location, we designed a network that outputs: (1) a global *kernel basis* $v_b[\delta, t]$, of size $K^2T \times B$ with $b \in \{1, \dots B\}$; and (2) a B dimensional coefficient vector $c_n[b]$ at each spatial location.

$$w_n[\delta, t] = \sum_b v_b[\delta, t] c_n[b]. \tag{3}$$

Note that we typically choose the number of basis kernels $B \ll K^2T$. This implies that all the kernels for a given burst lie in a low-dimensional subspace, but this subspace will be different for different bursts, *i.e.* the basis is burst-specific. This procedure allows us to recreate a full kernel field with far fewer predictions. Assuming a $W \times H$ resolution image, we need only make $WHB + K^2TB$ predictions to effectively recreate a kernel field of size WHK^2T.

We designed our network following an encoder-decoder architecture with skip connections [31]. Our model, however, has two decoder branches, one for the basis, the other for the coefficients (Figure 2). The encoder is shared between the two branches because the meaning of the coefficients c is dependent on the predicted basis v in Equation (3), so the two outputs need to be co-ordinated. This encoder takes the noisy burst and noisy parameters as input, and through multiple levels of downsampling and global average pooling at the end, yields a single global feature vector as its encoding of the image. The per-pixel coefficients c are then decoded from the encoder bottleneck to the full image resolution $W \times H$, with B channels as output. The common basis v is decoded up to *distinct* spatial dimensions — that of the kernels $K \times K$ — with $B \times T$ output channels.

Since the basis branch decodes to a different spatial resolution, we need a careful treatment of the skip connections. Unlike a usual U-Net, the encoder and decoder feature size do not match. Specifically, a pixel δ in the basis kernel $v_b[\delta, \cdot]$ has no meaningful relation to a pixel n in the input frames $I[n, \cdot]$. Therefore, in the skip connections from the shared encoder to the basis decoder, we apply a global spatial average pooling of the encoder's activations, and replicate the average vector to the resolution of the decoder layer. This mechanism ensures the encoder information is globally aggregated without creating nonsensical correspondences between kernel and image locations, while allowing features at multiple scales of the encoder to inform the basis decoder.

We ensure each of the reconstructed kernel w has positive weights that sum to one, to represent averaging. We implement this constraint using soft-max normalizations on *both* the coefficient and basis decoder outputs. So every 3D kernel of the basis $v_b[\cdot, \cdot]$ and every coefficient vector $c_n[\cdot]$ is normalized individually. A more detailed description of the architecture is provided in the supplementary.

Our network is trained with respect to the quality of the final denoised output $\hat{R}$—with an $L2$ loss on intensities and $L1$ loss on gradients. Like [27], we additionally use a per-frame loss to bias the network away from relying only on the reference frame. We do this with separate losses on denoised estimates from each individual frame of the input burst (formed as $\hat{R}_t[n] = T\sum_\delta w_n[\delta,t]I[n-\delta,t]$). These are added to main training loss, with a weight that is decayed across training iterations.

3.3. Efficient Fourier domain filtering

Filtering by convolution with large kernels is commonly implemented in the Fourier domain, where the filtering complexity is quasilinear in image size, while the complexity of direct convolution scales with the product of image- and kernel-size. But because the kernels w in KPNs vary spatially, Equation (2) does not represent a standard convolution, ruling out this acceleration.

In our case, because our kernels are defined with respect to a small set of "global" basis vectors, we can leverage Fourier-domain convolution to speed up filtering. We achieve this by combining and re-writing the expressions in Eq. (2) and Eq. (3) as:

$$\begin{aligned}\hat{R}[n] &= \sum_t\sum_\delta w_n[\delta,t]I[n-\delta,t]\\ &= \sum_t\sum_\delta\sum_b v_b[\delta,t]c_n[b]I[n-\delta,t]\\ &= \sum_b c_n[b]\sum_t\sum_\delta v_b[\delta,t]I[n-\delta,t]\\ &= \sum_b c_n[b]\sum_t \left(I[\cdot,t]\star v_b[\cdot,t]\right)[n], \end{aligned} \quad (4)$$

where $\star$ denotes standard spatial 2D convolution with a spatially-uniform kernel.

In other words, we first form a set of B filtered versions of the input burst I by standard convolution with each of the basis kernels—convolving each frame in the burst I with the corresponding "slice" of the basis kernel—and then taking a spatially-varying linear combination of the filtered intensities at each pixel based on the coefficients c. We can carry out these standard convolutions in the Fourier domain as:

$$I[\cdot,t]\star v_b[\cdot,t] = \mathcal{F}^{-1}\left(\mathcal{F}(I[\cdot,t])\cdot\mathcal{F}(v_b[\cdot,t])\right), \quad (5)$$

where $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ are spatial forward and inverse Fourier transforms. This is significantly more efficient for larger kernels, especially since we need not repeat forward Fourier transform of the inputs I for different basis kernels.

4. Experiments

We closely follow Mildenhall et al. [27] for training and evaluation.Our model is designed for bursts of $T = 8$ frames with resolution 128×128. Following the procedure of [27], we use training and validation sets constructed from the Open Images dataset [20], with shot and read noise parameters uniformly sampled in the log-domain: $\log(\sigma_r) \in [-3,-1.5]$, and $\log(\sigma_s) \in [-4,-2]$. We also use [27]'s set of 73 grayscale test images for evaluation.

Our default configuration uses bases with $B = 90$ kernels of size $K = 15$. We train our network (as well as all ablation baselines) using Adam [17] with a batch-size of 24 images, and an initial learning rate of 10^{-4}. We train for a total of about 600k iterations, dropping the learning twice, by $\sqrt{10}$ each time, whenever the validation loss saturates.

A reference implementation of our method is available at https://www.cse.wustl.edu/~zhihao.xia/bpn/.

Method	Gain $\propto 1$	Gain $\propto 2$	Gain $\propto 4$	Gain $\propto 8$
HDR+ [12]	31.96	28.25	24.25	20.05
BM3D [5]	33.89	31.17	28.53	25.92
NLM [2]	33.23	30.46	27.43	23.86
VBM4D [25]	34.60	31.89	29.20	26.52
Direct	35.93	33.36	30.70	27.97
KPN* [27]	36.47	33.93	31.19	27.97
KPN ($K = 5$)	36.35	33.69	31.02	28.16
MKPN [26]	36.88	34.22	31.45	28.52
BPN (ours)	**38.18**	**35.42**	**32.54**	**29.45**

Table 1: Denoising performance on a synthetic grayscale benchmark [27]. We report performance in terms of Average PSNR (dB). As in [27], our method was not trained on the noise levels implied by the largest gain (fourth column). KPN and MKPN refer to our implementation of these techniques, while numbers for KPN* correspond to those reported in the KPN paper itself [27]. Results for all other methods, including end-to-end regression to denoised intensities (denoted as Direct), are from [27]. Our method widely outperforms all prior methods at all noise levels.

4.1. Denoising performance

Table 1 reports the PSNR of our denoised outputs on the grayscale test set [27]. Each noise level corresponds to a sensor gain value (one stop increments of the ISO setting in a camera). The gains correspond to the following values for $(\log(\sigma_s),\log(\sigma_r))$: $1 \to (-2.2,-2.6), 2 \to (-1.8,-2.2), 4 \to (-1.4,-1.8), 8 \to (-1.1,-1.5)$. The highest noise level, denoted as Gain $\propto 8$, lies outside the range we trained on. We use it to evaluate our model's extrapolation capability. In addition to our own model, we also report results for a motion-alignment-based method [12], several approaches based on non-local filtering [2, 5, 25], as well as the standard KPN burst denoiser [27]—which is the current state-of-the-art. Since we did not have access to the original KPN model, we imple-

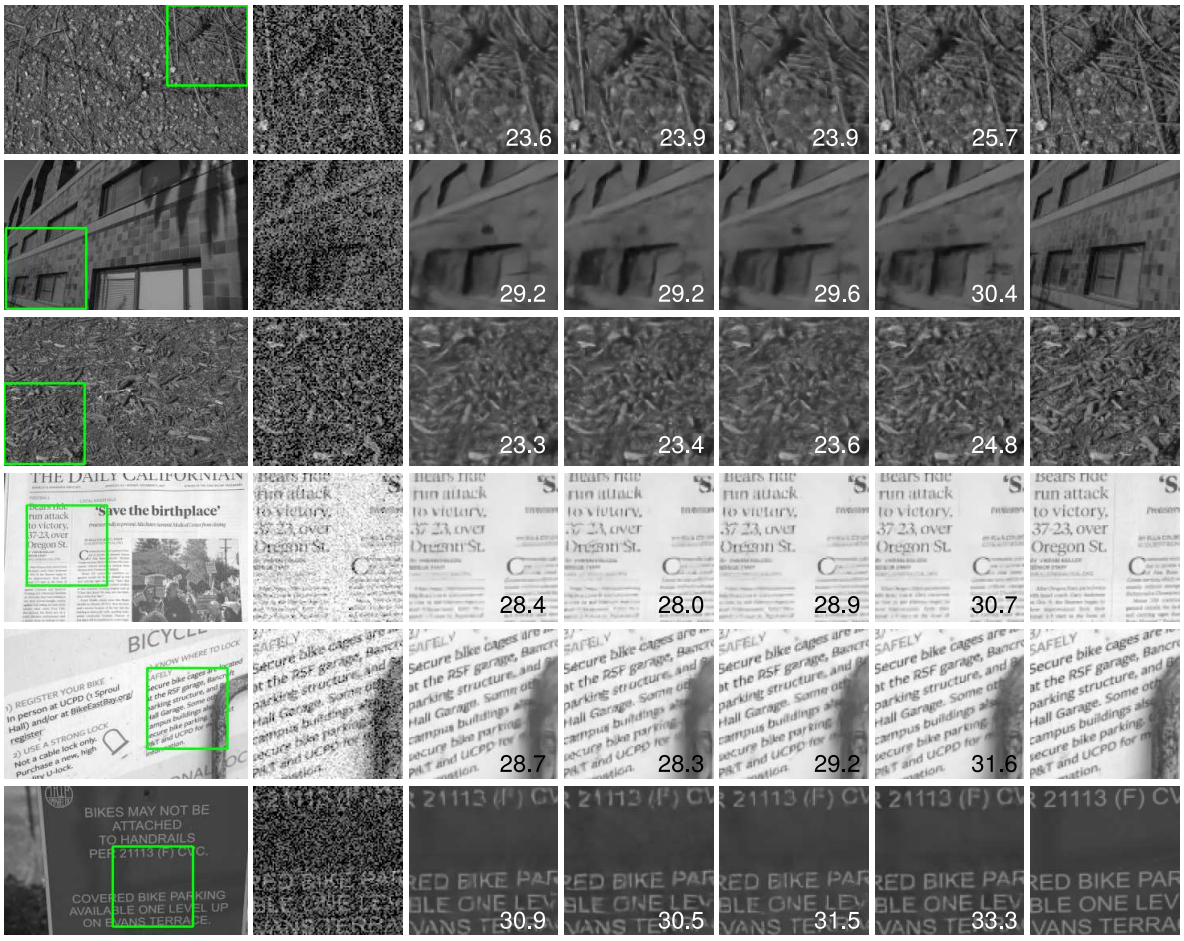

Figure 3: We illustrate denoising performance on a benchmark synthetic grayscale test set [27] for our method, a direct prediction network (which directly regresses denoised pixels), and two KPN variants [27, 26] with the same kernel size $K = 15$ as our method. The numbers in inset refer to the PSNR (dB) on the full image. In addition to better quantitative performance, our method does better at reproducing perceptual details like textures, edges, and text.

mented a version ourselves (that we use in ablations in the next section) and also report its performance in Table 1. We find it closely matches the performance reported in [27]. Additionally, we train a network to directly regress the denoised pixel values from the input burst (*i.e.*, without kernels), as well as our implementation of [26] with a larger kernel size of $K = 15$ for fair comparison.

We find that our method outperforms KPN [27] by a significant margin, over 1 dB PSNR at all noise levels. Our implementation of [26] also does well, but remains inferior to our model. We show qualitative results for a subset of methods in Figure 3. Our have fewer artifacts, especially in textured regions and around thin structures like printed text.

4.2. Ablation and analysis

Our approach leads to better denoising quality because it enables larger kernels without vastly increasing the network's output dimensionality and number of learnable parameters. To tease apart the contributions of kernel size and the structure of our kernel decomposition, we conduct an ablation study on our validation set. The results can be found in Table 2. The performance gap between the test and validation set results (Table 1 and 2) comes from differences in the datasets themselves.

Kernel Size. As a baseline, we consider using KPN directly with our larger kernel size of $K = 15$. We also consider predicting a single separable kernel at that size ([26] predicts separable kernels at multiple sizes, and adds them together). We find that our network outperforms the large kernel KPN variant at all noise levels—suggesting that simply increasing the kernel size is not enough. It also outperforms separable kernel prediction, suggesting that a low-dimensional subspace constraint better captures the structure of natural images than spatial separability.

	Gain $\propto 1$	Gain $\propto 2$	Gain $\propto 4$	Gain $\propto 8$
KPN ($K = 15$)	34.29	31.80	28.23	24.86
Separable ($K = 15$)	34.67	32.05	28.52	25.12
Ours ($K = 5$)	35.70	33.02	29.16	25.57
Ours ($K = 9$)	36.22	33.41	29.56	25.94
Ours ($B = 10$)	35.31	32.69	28.95	25.43
Ours ($B = 50$)	36.10	33.33	29.45	25.88
Ours ($B = 130$)	36.27	33.47	29.57	**25.99**
Ours ($\mathbf{K = 15, B = 90}$)	**36.29**	**33.57**	**29.62**	**25.99**
Common Spatial Basis	35.71	33.04	29.23	25.71
Per-frame Spatial Basis	36.21	33.46	29.56	25.92
Fixed basis	34.66	32.15	28.68	25.39

Table 2: Ablation study on our validation dataset. Performance is reported in terms of Average PSNR (dB). Beyond motivating our parameter choices ($K = 15$,$B = 90$), this demonstrates that our use of a burst-specific spatio-temporal basis outperforms standard KPN [27], separable spatial kernels, a common spatial basis for all burst frames, separate spatial bases per-frame, and a fixed, input-agnostic basis. All these variants were trained with the same settings ($K = 15$, $B = 90$) as our model.

For completeness, we also evaluate our basis prediction network with smaller kernels, $K = 9$ and $K = 5$. Although, this leads to a drop in performance compared to our default configuration, these variants still perform better than the original KPN—suggesting our approach has a regularizing effect that benefits even smaller kernels.

Basis Size. The number of basis elements in our default configuration, $B = 90$, was selected from a parameter search on the validation set. We include this analysis in Table 2, reporting PSNR values for B ranging from 10 to 130. We find that bases with fewer than 90 kernels lead to a drop in quality. The larger bases, $B = 130$, also performs very slightly worse than $B = 90$. We hypothesize that large bases start to have too many degrees of freedom. This increases the dimensionality of the network's output, which negates the benefits of a subspace restriction.

Spatial vs. Spatio-temporal Basis Decomposition. Note that we define our basis as a subspace to span 3D kernels—i.e., each of our basis elements v_b is a 3D spatio-temporal kernel. We predict a single weight $c_n[b]$ at each location, which is applied to corresponding spatial kernels $v_n[\cdot, t]$ for all frames t. However, there are other possible choices for decomposing 3D kernels, and we consider two of these in our ablation (Table 2). In both cases, we output coefficients $c_{n,t}[b]$ that vary per-frame, in addition to per-location—and are interpreted as separate coefficients corresponding to a spatial basis kernel. In one case, we use a *common spatial basis* $v_b[\delta]$ across all frames, with $w_n[\delta, t] = \sum_b c_{n,t}[b] v_b[\delta]$. In the other, we have a *per-frame spatial basis* $v_{b,t}[\delta]$ for each frame, and $w_n[\delta, t] = \sum_b c_{n,t}[b] v_{b,t}[\delta]$. The per-frame basis increases the dimensionality of our coefficient output and leads to a slight drop in performance, likely due to a reduced regularizing effect. The common spatial basis, however, suffers a greater performance drop since it also forces kernels in all frames to share the same subspace.

We also compare qualitatively the spatio-temporal kernels produced by our default configuration with those predicted by standard KPN in Figure 4. Our model makes better use of the temporal information, applying large weights to pixels across many frames in the burst, whereas KPN tends to overly favor the reference frame. Our network better tracks the apparent motion in the burst, shifting the kernel accordingly. And it is capable of ignoring outliers caused to excessive motion (all black kernels in Fig. 4).

Fixed vs. Burst-specific Basis. Given that our network predicts both a basis and per-pixel coefficients, a natural question is whether a burst-specific kernel basis is even needed. To address this, we train a network architecture without a basis decoder to only predict coefficients for each burst, and instead learn a basis that is fixed across all bursts in the training set. The fixed basis is learned jointly with this network as a direct learnable tensor. Table 2 shows that using a fixed basis in this manner leads to a significant decrease in denoising quality (although still better than standard KPN).

This suggests that while a subspace restriction on kernels is useful, the ideal subspace is scene-dependent and must be predicted adaptively. We further explore this phenomenon in Table 3, where we quantify the rank of the predicted bases for individual images, and for pairs of images. Note that the rank can be lower than B, since we do not effectively require the 'basis' vectors $\{v_b[\cdot, \cdot]\}$ to be linearly independent. We find that the combined rank of basis kernels of image pairs (obtained by concatenating the two bases) is nearly twice the rank obtained from individual images—suggesting limited overlap between the basis sets of different images. We also explicitly compute the average overlap ratio across image pairs as $1 - rank(v, v')/[rank(v) + rank(v')]$, and find it to be around 5% on average. This low overlap implies that different bursts do indeed require different bases, justifying our use of burst-specific bases.

4.3. Computational expense

Next, we evaluate the computational expense of our approach and compare it to the different ablation settings considered in Table 2, including standard KPN. We report the total number of floating point operations (FLOPs) required for network prediction and filtering in Table 4. We find that in addition to producing higher-quality results, our approach also requires significantly fewer FLOPs than regular KPN for the same kernel size. This is due to the reduced

	Gain $\propto 1$	Gain $\propto 2$	Gain $\propto 4$	Gain $\propto 8$
$rank(v)$	80.6	81.8	84.3	86.2
$rank(v, v')$	152.2	154.5	159.6	165.2
Overlap ratio	5.4%	5.5%	5.4%	4.4%

Table 3: Average basis rank for each noise level (first row), average rank of the union of two bases from random burst pairs (second row), and the average overlap ratio (third row) between the subspaces spanned by the two bases. The low overlap justifies our prediction of a burst-specific basis.

	GFLOPs	Runtime (s)
KPN ($K = 15$)	59.3	0.63
Separable ($K = 15$)	29.9	0.43
Ours ($K = 5$)	28.9	0.24
Ours ($K = 9$)	29.1	0.29
Ours ($B = 10$)	26.5	0.19
Ours ($B = 50$)	28.2	0.27
Ours ($B = 130$)	31.7	0.41
Ours ($\mathbf{K = 15, B = 90}$)	29.9	0.30
Common Spatial Basis	40.8	0.49
Per-frame Spatial Basis	41.9	0.57

Table 4: FLOPS and runtimes on 1024×768 resolution images for different KPN denoising approaches. All variants of our basis prediction network are significantly faster than KPN and match the compute cost of separable filters (with better denoising quality). Increasing the kernel size for our technique comes at marginal cost thanks for the Fourier filtering approach. This allows us to use large kernels for better denoising performance.

complexity of our final prediction layer, as well as efficient filtering in the Fourier domain. Also, we find that our approach has nearly identical complexity as separable kernel prediction, while achieving higher denoising performance because it can express a more general class of kernels.

In addition to the evaluation FLOPs, Table 4 reports measured running times for the various approaches, benchmarked on a 1024 × 768 image on an NVIDIA 1080Ti GPU. To compute these timings, we divide the image into 128×128 non-overlapping patches to form a batch and send it to the denoising network. Since regular KPN has high memory requirements, we select the maximum batch size for each method and denoise the entire image in multiple runs. This maximizes GPU throughput. We find that our approach retains its running time advantage over KPN in practice. It is also a little faster than separable kernel prediction—likely due to the improved cache performance we get from using Fourier-domain convolutions.

Method	Gain $\propto 1$	Gain $\propto 2$	Gain $\propto 4$	Gain $\propto 8$
Direct	38.16	35.39	32.50	30.27
KPN ($K = 5$)	38.86	35.97	32.79	30.01
BPN (ours)	**40.16**	**37.08**	**33.81**	**31.19**

Table 5: Denoising performance on our synthetic color test set. We report performance in terms of Average PSNR (dB). Here, KPN refers to our extension of [27] to produce color kernels. Our method outperforms KPN by more than 1 dB at all noise levels.

4.4. Color burst denoising

Finally, we report results on a color burst denoising task. We use a similar observation model as (1) with noise independently added to each color channel (note this ignores multiplexed measurements and demosaicking). For kernel denoising, we use separate kernels for each color channel at each location. We extend standard KPN [27] to produce this directly, and modify our method to have the basis decoder produce a "color" kernel basis (of size $3K^2T \times B$), while the coefficient decoder still outputs a B dimensional coefficient vector.

We use the same training protocol as for grayscale images, using color versions of the Open Images dataset. In this case, training takes 1900k iterations with batches of 8 color images. We construct a new synthetic test set of 100 images from the Open Images validation dataset, with no overlap with our training set. We report comparisons in Table 5, showing a similar improvement over KPN [27] as for grayscale images—over 1 dB PSNR at all noise levels. We include qualitative comparisons in Figure 5.

5. Conclusion and future work

In this work, we argue that local, per-pixel burst denoising kernels are highly coherent. Based on this, we present a basis prediction network that jointly infers a global, low-dimensional kernel basis and the corresponding per-pixel mixing coefficients that can be used to construct per-pixel denoising kernels. This formulation significantly reduces memory and compute requirements compared to prior kernel-predicting burst denoising methods, allowing us to substantially improve performance by using large kernels, while reducing running time.

While this work focuses on burst denoising, KPN-based methods have been applied to other image and video enhancement tasks including video super-resolution [16], frame interpolation [28, 29, 23], video prediction [15, 6], and video deblurring [41]. All these tasks exhibit similar structure and will likely benefit from our approach.

There are other forms of spatiotemporal structure that can be explored to build on our work. For example, image

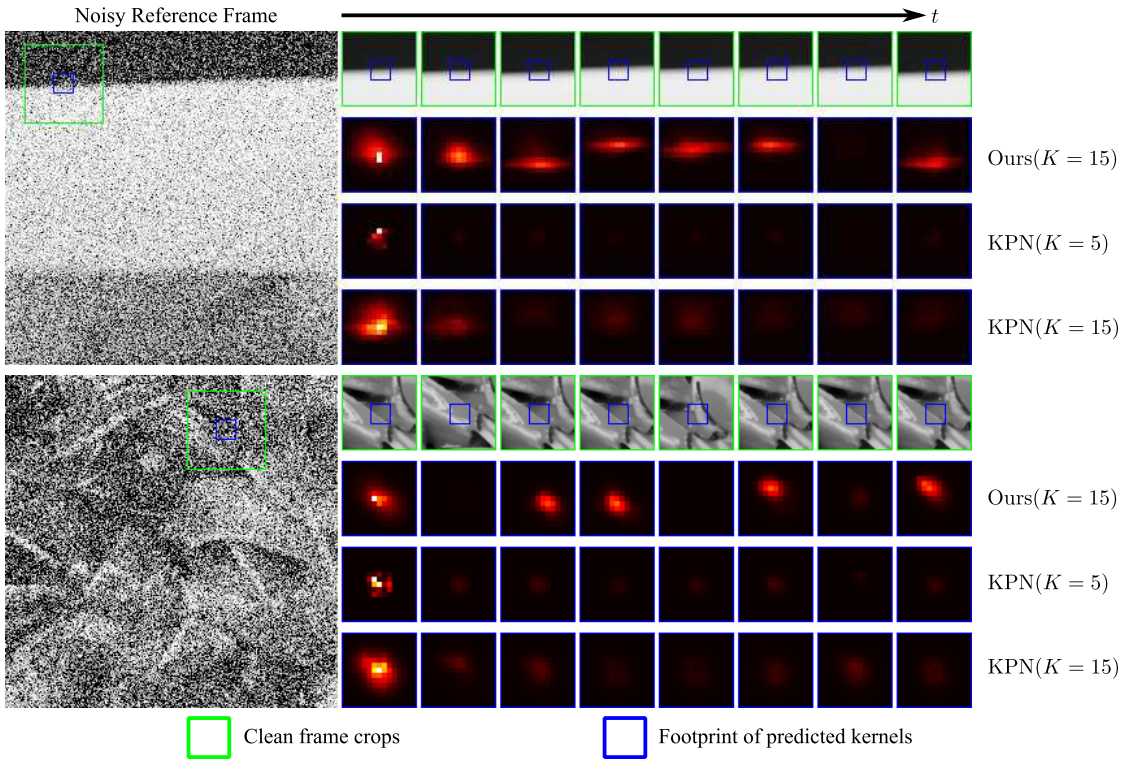

Figure 4: We visualize a few 3D kernels predicted by our approach (with $K = 15$), and those produced by standard KPN (with $K = 5$ and $K = 15$). For kernels predicted at a given location, we also show crops of the different noise-free frames centered at that point, with the support of the kernel marked in blue. In comparison to those from KPN, our kernels are more evenly distributed across all frames in the burst, with spatial patterns that closely follow the apparent motion in the burst.

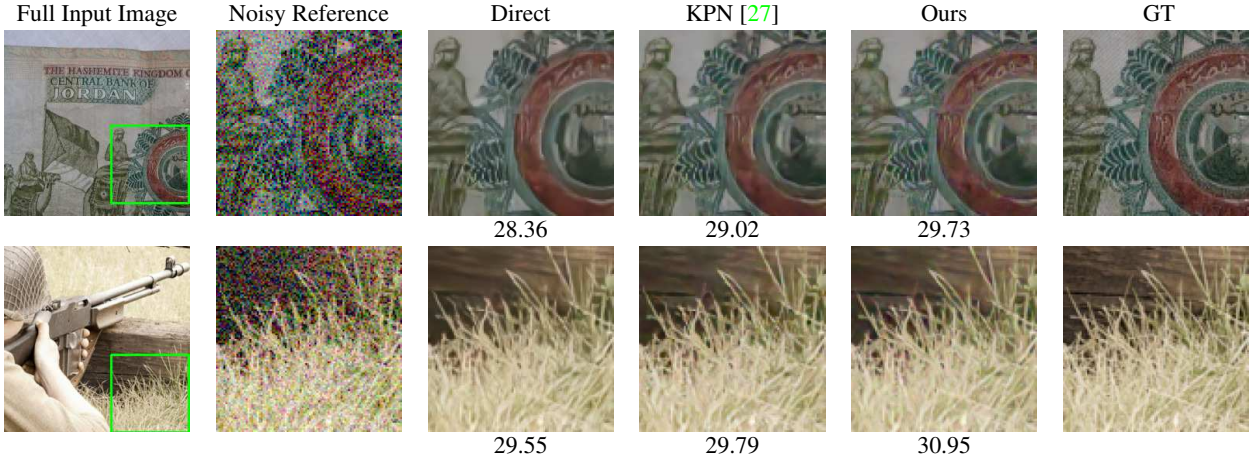

Figure 5: We show examples of color denoising using our method on our synthetic color test set, comparing these to direct prediction and our color-extended version of KPN [27] (with $K = 5$). Numbers refer to PSNR (dB) on the full image.

enhancement methods have exploited self-similarity at different scales [10] suggesting other decompositions in scale space. Also, we assume a fixed basis size globally. Adapting this spatially to local content could yield further benefits. Finally, KPNs are still, at their heart, local filtering methods and it would be interesting to extend our work to non-local filtering methods [5, 19, 25].

Acknowledgments. ZX and AC acknowledge support from NSF award IIS-1820693, and a gift from Adobe Research.

References

[1] Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony Derose, and Fabrice Rousselle. Kernel-predicting convolutional networks for denoising monte carlo renderings. *ACM Transactions on Graphics (TOG)*, 36(4):97, 2017.

[2] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *Proc. CVPR*, 2005.

[3] Harold C Burger, Christian J Schuler, and Stefan Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *Proc. CVPR*, 2012.

[4] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *Proc. CVPR*, 2018.

[5] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Color image denoising via sparse 3d collaborative filtering with grouping constraint in luminance-chrominance space. In *Proc. ICIP*, 2007.

[6] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, pages 64–72, 2016.

[7] 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.

[8] 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.

[9] Michaël Gharbi, Tzu-Mao Li, Miika Aittala, Jaakko Lehtinen, and Frédo Durand. Sample-based monte carlo denoising using a kernel-splatting network. *ACM Transactions on Graphics (TOG)*, 38(4):125, 2019.

[10] Daniel Glasner, Shai Bagon, and Michal Irani. Super-resolution from a single image. In *ICCV*, 2009.

[11] Clément Godard, Kevin Matzen, and Matt Uyttendaele. Deep burst denoising. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 538–554, 2018.

[12] Samuel W Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T Barron, Florian Kainz, Jiawen Chen, and Marc Levoy. Burst photography for high dynamic range and low-light imaging on mobile cameras. *ACM Transactions on Graphics (TOG)*, 35(6):192, 2016.

[13] Felix Heide, Steven Diamond, Matthias Nießner, Jonathan Ragan-Kelley, Wolfgang Heidrich, and Gordon Wetzstein. Proximal: Efficient image optimization using proximal algorithms. *ACM Transactions on Graphics (TOG)*, 35(4):84, 2016.

[14] Felix Heide, Markus Steinberger, Yun-Ta Tsai, Mushfiqur Rouf, Dawid Pająk, Dikpal Reddy, Orazio Gallo, Jing Liu, Wolfgang Heidrich, Karen Egiazarian, et al. Flexisp: A flexible camera image processing framework. *ACM Transactions on Graphics (TOG)*, 33(6):231, 2014.

[15] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. In *Advances in Neural Information Processing Systems*, pages 667–675, 2016.

[16] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3224–3232, 2018.

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

[18] Filippos Kokkinos and Stamatis Lefkimmiatis. Iterative residual cnns for burst photography applications. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5929–5938, 2019.

[19] Dabov Kostadin, Foi Alessandro, and Egiazarian Karen. Video denoising by sparse 3d transform-domain collaborative filtering. In *European signal processing conference*, volume 149, 2007.

[20] Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*, 2017.

[21] Orly Liba, Kiran Murthy, Yun-Ta Tsai, Tim Brooks, Tianfan Xue, Nikhil Karnad, Qiurui He, Jonathan T. Barron, Dillon Sharlet, Ryan Geiss, Samuel W. Hasinoff, Yael Pritch, and Marc Levoy. Handheld mobile photography in very low light. *ACM Trans. Graph.*, 38(6):164:1–164:16, Nov. 2019.

[22] Ding Liu, Bihan Wen, Yuchen Fan, Chen Change Loy, and Thomas S Huang. Non-local recurrent network for image restoration. In *Advances in Neural Information Processing Systems*, pages 1680–1689, 2018.

[23] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4463–4471, 2017.

[24] Ziwei Liu, Lu Yuan, Xiaoou Tang, Matt Uyttendaele, and Jian Sun. Fast burst images denoising. *ACM Transactions on Graphics (TOG)*, 33(6):232, 2014.

[25] 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.

[26] Talmaj Marinč, Vignesh Srinivasan, Serhan Gül, Cornelius Hellge, and Wojciech Samek. Multi-kernel prediction networks for denoising of burst images. *arXiv preprint arXiv:1902.05392*, 2019.

[27] Ben Mildenhall, Jonathan T Barron, Jiawen Chen, Dillon Sharlet, Ren Ng, and Robert Carroll. Burst denoising with kernel prediction networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2502–2510, 2018.

[28] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 670–679, 2017.

[29] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 261–270, 2017.

[30] Yaniv Romano, John Isidoro, and Peyman Milanfar. Raisr: rapid and accurate image super resolution. *IEEE Transactions on Computational Imaging*, 3(1):110–125, 2016.

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241, 2015.

[32] Stefan Roth and Michael J Black. Fields of experts. *IJCV*, 2009.

[33] Uwe Schmidt and Stefan Roth. Shrinkage fields for effective image restoration. In *Proc. CVPR*, 2014.

[34] Thijs Vogels, Fabrice Rousselle, Brian McWilliams, Gerhard Röthlin, Alex Harvill, David Adler, Mark Meyer, and Jan Novák. Denoising with kernel prediction and asymmetric loss functions. *ACM Transactions on Graphics (TOG)*, 37(4):124, 2018.

[35] Zhihao Xia and Ayan Chakrabarti. Identifying recurring patterns with deep neural networks for natural image denoising. *arXiv preprint arXiv:1806.05229*, 2018.

[36] Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. In *NeurIPS*, pages 341–349, 2012.

[37] Tianfan Xue, Jiajun Wu, Katherine Bouman, and Bill Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in neural information processing systems*, pages 91–99, 2016.

[38] 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.

[39] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *Proc. CVPR*, 2017.

[40] Yulun Zhang, Kunpeng Li, Kai Li, Bineng Zhong, and Yun Fu. Residual non-local attention networks for image restoration. *arXiv preprint arXiv:1903.10082*, 2019.

[41] Shangchen Zhou, Jiawei Zhang, Jinshan Pan, Haozhe Xie1, Wangmeng Zuo, and Jimmy Ren. Spatio-temporal filter adaptive network for video deblurring. In *Proc. ICCV*, 2019.

[42] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *Proc. ICCV*, 2011.