

Non-uniform Blur Kernel Estimation via Adaptive Basis Decomposition

Guillermo Carbajal
Universidad de la República
Uruguay

Patricia Vitoria
Universitat Pompeu Fabra
Spain

Mauricio Delbracio
Universidad de la República
Uruguay

Pablo Musé
Universidad de la República
Uruguay

José Lezama
Universidad de la República
Uruguay

Abstract

Motion blur estimation remains an important task for scene analysis and image restoration. In recent years, the removal of motion blur in photographs has seen impressive progress in the hands of deep learning-based methods, trained to map directly from blurry to sharp images. Characterization of the motion blur, on the other hand, has received less attention, and progress in model-based methods for deblurring lags behind that of data-driven end-to-end approaches. In this work we revisit the problem of characterizing dense, non-uniform motion blur in a single image and propose a general non-parametric model for this task. Given a blurry image, a neural network is trained to estimate a set of image-adaptive basis motion kernels as well as the mixing coefficients at the pixel level, producing a per-pixel motion blur field. We show that our approach overcomes the limitations of existing non-uniform motion blur estimation methods and leads to extremely accurate motion blur kernels. When applied to real motion-blurred images, a variational non-uniform blur removal method fed with the estimated blur kernels produces high-quality restored images. Qualitative and quantitative evaluation shows that these results are competitive or superior to results obtained with existing end-to-end deep learning (DL) based methods, thus bridging the gap between model-based and data-driven approaches.

1. Introduction

Motion blur is a major source of degradation in digital images. This effect, preeminent in low light photography, occurs when the exposure time is large compared to the relative motion speed between the camera and the scene. As a result, the camera sensor at each pixel receives and accumulates light coming from different sources, producing different amounts of blur.

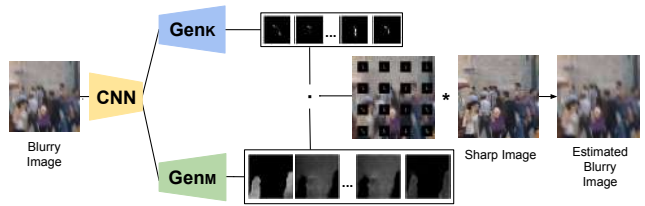


Figure 1: **Overview of the proposed method.** Given a blurry image, a neural network predicts a set of kernel basis and corresponding pixel-wise mixing coefficients allowing to reblur the corresponding sharp image by first convolving it with each basis kernel and performing a weighted sum using the mixing coefficients.

Single image blur kernel estimation is an ill-posed problem, as many pairs of blur kernels and images can generate the same blurry image. In model-based methods, an accurate blur kernel estimation is important for obtaining a high-quality sharp image. Existing non-uniform methods approximate the blur kernel by assuming a parametric model of the motion field, either by considering a global parametric form induced by camera motion [6, 8, 27, 34], or by locally modeling the motion field with linear kernels parameterized by the length and orientation of the kernel support [5, 12, 27, 29]. While these methods reduce significantly the complexity and computational cost by solving for a simple approximate motion field, the approximation does not generalize to most real case scenarios, like camera shake from hand tremor [4].

Motion blur kernel estimation proves useful to infer scene motion information and to solve related tasks such as tracking and motion segmentation. However, its major motivation is certainly blind motion deblurring. Recently, the unprecedented improvements in deblurring obtained by DL methods have contributed to a decreased interest in model-based approaches and motion blur kernel estimation. In-

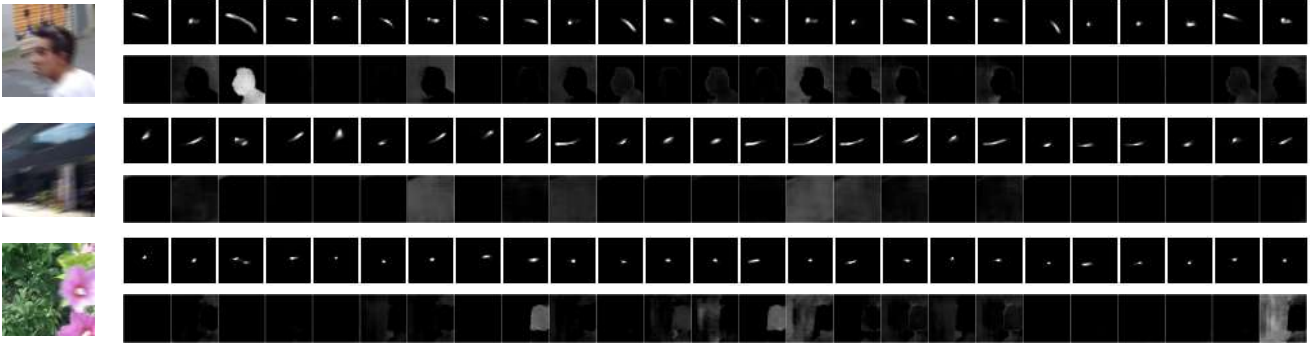


Figure 2: **Examples of generated kernel basis $\{k^b\}$ and corresponding mixing coefficients $\{m^b\}$** predicted from the blurry images shown on the left. The adaptation to the input is more notorious for the elements that have significant weights.

deed, DL deblurring methods skip the kernel estimation step arguing that [20]: (i) blur kernels models are too simplistic and unrealistic to be used in practice; (ii) the kernel estimation process is subtle and sensitive to noise and saturation; (iii) finding a spatially varying kernel for every pixel in a dynamic scene requires a huge amount of memory and computation. While these limitations are certainly true, current deblurring DL-based methods also suffer from a major limitation related to the way the training datasets are produced. To synthesize motion-blurred images without relying on a forward kernel-based model, the training dataset is obtained by averaging consecutive frames from a sequence of a dynamic scene captured with a high-speed camera [19, 28, 29]. Such datasets are only able to characterize the motion blur to a limited extent, and their end-to-end training schemes induce a mapping that might be specific to the camera that was used, capturing transformations other than deblurring. As a consequence, and as confirmed by the experiments presented in this work, DL-based methods sometimes fail to generalize to real blurred images.

In this work, we propose a novel approach for non-parametric, dense, spatially-varying motion blur estimation based on an efficient low-rank representation of the per-pixel motion blur kernels. More precisely, for each blurred image, a convolutional neural network (CNN) estimates an image-specific set of kernel basis functions, as well as a set of pixel-wise mixing coefficients, cf. Figures 1, 2, 9. The combination of the two results in a unique motion blur kernel per pixel. Additionally, our model can handle saturation regions by explicitly modeling this phenomenon. When applied to image deblurring, this model leads to high-quality images whether or not saturated regions exist. We show that the proposed method is capable to overcome the above-mentioned limitations of model-based methods, producing precise spatially varying kernels while reducing the complexity of the problem.

We design the following strategy to assess the quality

of the estimated motion blur kernels: (i) We compare the estimated kernels obtained by our method with the ones obtained by prior art [5, 29]; (ii) We apply the classical Richardson-Lucy [17, 25] deblurring method to the non-uniform case, and compare the results using our estimated motion blur fields with state-of-the-art DL-based end-to-end methods [14, 15, 26, 31, 38]. This comparison shows not only that our motion blur kernels are extremely accurate, but that the motion deblurring obtained as a by-side product is competitive and is often able to outperform DL-based methods in standard benchmarks of real blurred images. Code and pre-trained model weights are publicly available¹.

2. Related Work

Single Image Non-Uniform Motion Blur Estimation.

The literature on motion kernel estimation is extremely vast. We limit the current analysis to spatially varying kernel estimation methods from a single image. Early methods attempting to estimate non-uniform motion blur kernels are limited to the case of camera egomotion [6, 8, 30, 34]. They assume the scenes are planar and therefore are mapped to the image plane by a homography. To deal with spatially-varying blur due to depth and moving objects, methods like [11, 23] propose to segment the image in a reduced number of layers at fixed depth. One drawback of the piecewise constant model is that it can be sensitive to the segmentation of the blurred image.

A different approach, more related to ours, that deals with both scene depth variations and moving objects, consists of predicting motion blur locally. Sun *et al.* [29] consider a set of pre-defined linear motion kernels parameterized by their lengths and orientations. Then, they predict a patch-level probabilistic distribution of the kernel parameters using a CNN. The patch-level distribution is then converted to a dense motion field using a Markov random field

¹<https://github.com/GuillermoCarbajal/NonUniformMotionBlurKernelEstimation>

enforcing motion smoothness.

Given a blurry image, Gong *et al.* [5] directly estimate a dense linear motion flow parameterized by their horizontal and vertical components using a Fully Convolutional Network (FCN). To train the FCN, they simulate motion flows to generate synthetic blurred image/motion flow pairs. Both methods propose to deblur images using a non-blind deblurring method with the estimated kernels, combining an L^2 data fitting term with EPLL image prior [41].

Kernel Prediction Networks. Recently, Kernel Prediction Networks (KPN) have been proposed for low-level vision tasks such as burst denoising [18, 35], optical flow estimation, frame interpolation [21, 22], stereo and video prediction [10]. Several works have used KPNs in the context of burst denoising. Mildenhall *et al.* [18] produce denoised estimates at each pixel as a weighted average of observed local noisy pixels across all the input frames. These averaging weights, or kernels, are predicted from the noisy input burst using a CNN. To improve the computational efficiency, Xia *et al.* [35] propose a 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.

Relation with the Proposed Approach. In this work, we propose a dense non-uniform motion blur estimation based on an efficient low-rank representation of the pixel-wise motion blur kernels. As in [6, 8, 34] the per-pixel kernels are modeled as linear combinations of base elements, but instead of having a single pre-computed basis of elements, we use a KPN to infer a non-parametric basis specific for each input image. Additionally, instead of learning the kernels to solve the image restoration problem (e.g. denoising in [35]), we learn them to fit the forward *degradation* model, enabling the utilization of existing techniques for solving inverse problems.

3. Proposed Motion Blur Degradation Model

Non-uniform motion blur can be modeled as the local convolution of a sharp image with a spatially varying filter, the *motion blur field*. This simple model represents the integration, at each pixel, of photons arriving from different sources due to relative motion between the camera and the scene. Given a sharp image \mathbf{u} of size $H \times W$, and a set of per-pixel blur kernels \mathbf{k}_i of size $K \times K$, we will assume that the observed blurry image \mathbf{v} is generated as

$$v_i = \langle \mathbf{u}_{nn(i)}, \mathbf{k}_i \rangle + n_i, \quad (1)$$

where $\mathbf{u}_{nn(i)}$ is a window of size $K \times K$ around pixel i in image \mathbf{u} and n_i is additive noise. We assume that kernels are non-negative (no negative light) and of area one (conservation of energy).

Predicting the full motion field of per-pixel kernels \mathbf{k}_i would lead to an estimation problem in a very high-dimensional space (K^2HW), being computationally intractable for large images and kernels. We propose an efficient solution, based on the assumption that there exists significant redundancy between the kernels present in the image. We incorporate this assumption in our model by imposing a low-rank structure to the field of motion kernels. We decompose the per-pixel kernels as linear combinations of elements in a much smaller basis. Specifically, if the blur kernel basis has B elements, then, only B coefficients \mathbf{m}_i^b are required per pixel instead of the original $K \times K$. Additionally, the B basis elements need to be estimated leading to an estimation problem of dimension $B(K^2 + HW)$. A notorious gain is obtained when $B \ll K^2$, particularly relevant for large blur kernels. The mixing coefficients are normalized so that they sum to one at each pixel location. Thus, the per-pixel kernel \mathbf{k}_i results from the convex combination of the basis kernel, conservation of energy is guaranteed, and the degradation model becomes:

$$v_i = \langle \mathbf{u}_{nn(i)}, \sum_{b=1}^B \mathbf{k}^b m_i^b \rangle + n_i. \quad (2)$$

Taking into account the sensor saturation, and the gamma correction performed by the camera, a more accurate model is given by

$$v_i = R(\langle \mathbf{u}_{nn(i)}, \sum_{b=1}^B \mathbf{k}^b m_i^b \rangle + n_i)^{\frac{1}{\gamma}}, \quad (3)$$

where γ is the gamma correction coefficient and $R(\cdot)$ is the pixel saturation operator that clips image values v_i which are larger than 1. To avoid the non-differentiability of this function at $v_i = 1$, a smooth approximation for the captor response function is considered [33]:

$$R(v_i) = v_i - \frac{1}{a} \log(1 + e^{a(v_i-1)}). \quad (4)$$

The parameter a controls the smoothness of the approximation and is set to $a = 50$. As for the gamma correction factor, a typical value of $\gamma = 2.2$ is used.

4. Synthetic Dataset Generation

We build a synthetic dataset consisting of 5,888 images from the ADE20K semantic segmentation dataset [40]. To generate random motion kernels, we use a camera-shake kernel generator [4, 2] based on physiological hand tremor data and pre-compute 500,000 kernels with support smaller than $K \times K$, cf. Figure 5. In our experiments, we observed that training on this synthetic data generalizes remarkably well to real photographs with different types of scenes and motion.

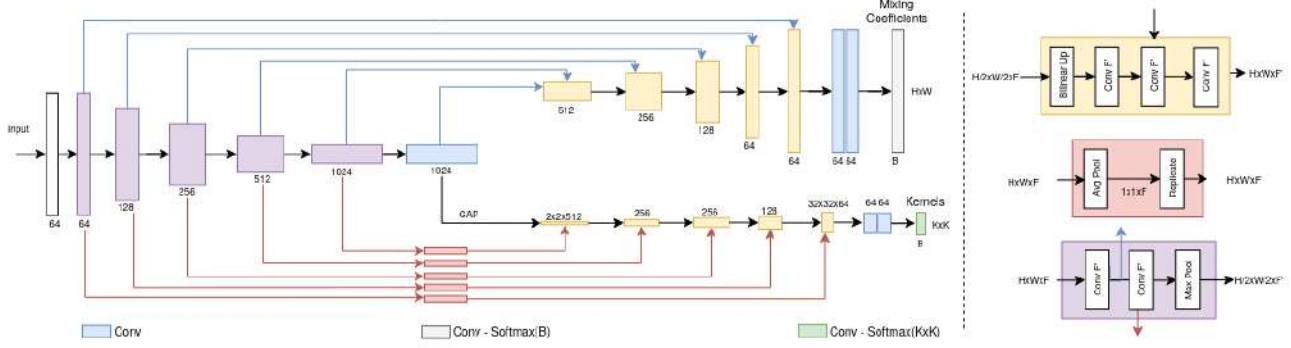


Figure 3: **Architecture Details** The proposed network is composed by an encoder and two decoders. The encoder takes as an input a blurry image. The decoders output the motion kernel basis and corresponding per-pixel mixing coefficients.

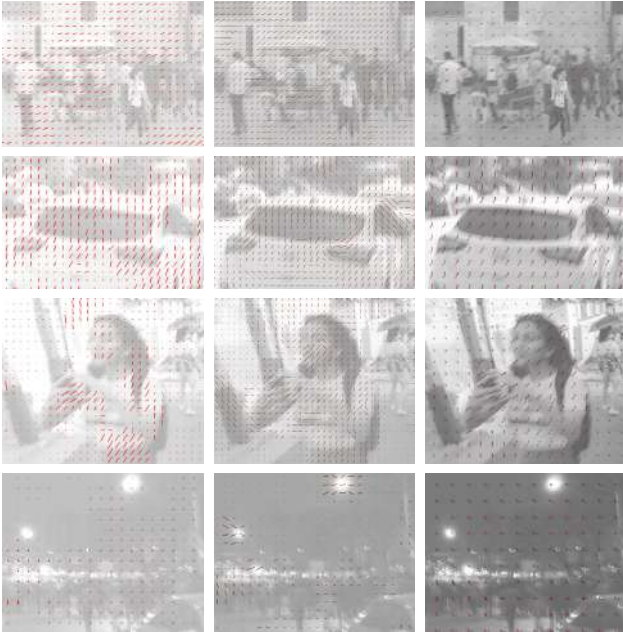


Figure 4: **Visual comparison of non-uniform motion blur kernel estimation.** From left to right: Gong *et al.* [5], Sun *et al.* [29] and ours. From top to bottom: two examples from the GoPro dataset [20], one from REDs [19] and one Lai’s dataset [16]. Best viewed in electronic format.

More specifically, for a random sharp image \mathbf{u} , we perform a convolution of the image with a random kernel \mathbf{k} . Additionally, each segmented object (if any) and its corresponding mask is convolved with a different random kernel. This ensures a soft transition between different blurry regions. To simplify, a maximum of three segmentation masks with a minimum size of 400 pixels are considered for each image. Finally, for each image, we obtain a tuple $(\mathbf{u}^{GT}, \mathbf{v}^{GT}, \{\mathbf{k}\}^{GT}, \{\mathbf{m}\}^{GT})$ containing the sharp image, blurry image and the pairs of ground truth kernels and

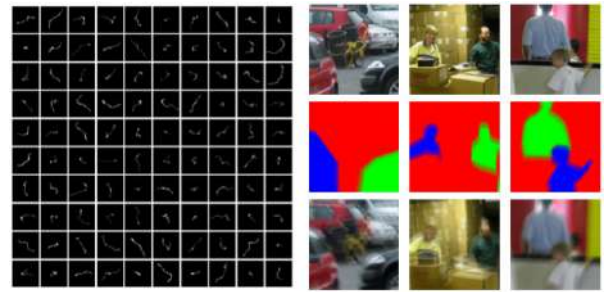


Figure 5: **Synthetic dataset for learning non-uniform motion kernel estimation.** Left: example random kernels. Right: different segments of the scene are convolved with different random kernels and are mixed with the (also convolved) segmentation masks.

masks applied to generate the blurry image.

Data Augmentation. To simulate latent scenes with saturating light sources, sharp images are first converted to *hsv* color space and then the histogram is transformed by multiplying the *v*-channel by a random value between $[0.5, 1.5]$. The transformed image is converted back to the *rgb* color space, and the blurry image is generated as described above. Finally, the blurry image is clipped to the $[0, 1]$ range. We refer to Section A.3 for more details and examples from our dataset.

5. Motion Blur Field Estimation

We use a CNN to estimate, from a given input blurry image, both the B basis motion kernels $\{\mathbf{k}^b\}_{b=1,\dots,B}$ and the mixing coefficients $\{\mathbf{m}^b\}_{b=1,\dots,B}$. Building upon recent work in kernel prediction networks [35], the network is composed of a shared backbone and two generator heads. We refer to Figure 3 for an overview of the architecture. Normalization of the blur kernels and the mixing coefficients is achieved by using Softmax layers. In our exper-

iments, we used $K = 33$ and $B = 25$. An ablation study exploring different values of B is given in Table 4.

5.1. Objective Function

Reblur Loss. Given a blurry image \mathbf{v}^{GT} , we aim to find the global kernel basis $\{\mathbf{k}^b\}$ and mixing coefficients $\{\mathbf{m}^b\}$ that minimize

$$\mathcal{L}_{reblur} = \sum_i w_i (v_i - v_i^{GT})^2, \quad (5)$$

where the v_i are computed using (3). The re-blurring of the sharp image can be done efficiently by first convolving it with each of the kernels in the base, and then doing an element-wise blending of the B resulting images with the corresponding mixing coefficients. To prevent a single kernel from dominating the loss, weights w_i are computed as the inverse of the number of pixels that belong to the same segmented object.

Kernel Loss. Ground truth pixel-wise motion blur kernels are compared to the predicted per-pixel kernels. Given a ground truth per-pixel blur kernel $\{\mathbf{k}_i^{GT}\}$, the computed kernel basis $\{\mathbf{k}^b\}$ and mixing coefficients $\{\mathbf{m}_i^b\}$, the *kernel loss* is defined as:

$$\mathcal{L}_{kernel} = \sum_i w_i \left\| \sum_{b=1}^B m_i^b \mathbf{k}^b - \mathbf{k}_i^{GT} \right\|_p, \quad (6)$$

where the weights w_i are the same as in *reblur loss*, and p is 1 or 2, depending on the training strategy described next.

5.2. Model Training

We train the CNN using the sum of the *reblur loss* (5) and the *kernel loss* (6) with equal weights. Training a model to predict a per-pixel kernel estimation is a difficult task. In our case, the model needs to figure out an image-specific low-rank decomposition to approximate all the kernels present in the image. In our experiments we observed very slow convergence and only started to see well-shaped kernels after around 200 epochs. We found that using an L^2 -norm on the kernels loss in (6) was adequate to find a first approximation of the model. After 300 epochs, we switched to the more robust L^1 -norm, which is harder to optimize but allows us to recover sharper kernels. In total, we trained our model for 1200 epochs using image patches of 256×256 pixels. Additional details on the training procedure and hyper-parameters are given in Section A.2.

6. Qualitative Evaluation

In Figures 2 and 9, we show examples of the set of kernel basis and corresponding mixing coefficients predicted for different images. Note that the predicted basis is image-dependent, specially for those kernel basis elements that are

more active in the decomposition (i.e. the corresponding mixing coefficients have high values throughout the scene).

Figure 4 shows some examples of non-uniform blur kernel estimates obtained by our method. We visually compare them with the results of two other existing DL-based non-uniform motion blur estimation methods, Gong *et al.* [5] and Sun *et al.* [29]. Despite being trained on synthetically blurred images, our method generalizes remarkably well to real blurry images (last column), as well as blurry images synthesized from video sequences as in GoPro [20] and REDs [19] datasets.

Our model is able to characterize different types of camera and object motions. Note also that the motion blur kernels estimated by the compared methods tend to correlate with the scene geometry, instead of revealing the underlying motion field. Moreover, our model predicts continuous free-form arbitrary motion kernels, whereas [29] and [5] are restricted to linear ones.

7. Application to Image Deblurring

In this section, we demonstrate the use of our estimated motion kernels for solving the inverse problem of motion blur removal in real photographs.

7.1. Problem Formulation and Algorithm

The estimated per-pixel kernels form the *forward model* of the blurring operation and can be used with any general linear inverse problem solver. Here we employ the classical Richardson-Lucy (RL) algorithm [30]. In its basic form, RL recovers the latent sharp image as the *maximum-likelihood* estimate under a *Poisson noise* model. Following the notation in Section 3, the likelihood of the blurry image \mathbf{v} as a function of the latent image \mathbf{u} can be written as

$$p(\mathbf{u}|\mathbf{v}) = \prod_i \frac{\hat{v}_i^{v_i} e^{-\hat{v}_i}}{v_i!}, \quad (7)$$

where

$$\hat{v}_i = \langle \mathbf{u}_{nn(i)}, \mathbf{k}_i \rangle = \langle \mathbf{u}_{nn(i)}, \sum_{b=1}^B \mathbf{k}^b m_i^b \rangle = (\mathbf{H}\mathbf{u})_i, \quad (8)$$

being \mathbf{H} the *degradation* matrix of size $N_v \times N_v$ for a N_v pixels image. Each row in matrix \mathbf{H} contains the corresponding per-pixel kernel. Maximizing (7) subject to the non-negativity constraint on \mathbf{u} gives rise to the RL update:

$$\hat{\mathbf{u}}^{t+1} = \hat{\mathbf{u}}^t \circ \mathbf{H}^T \left(\frac{\mathbf{v}}{\mathbf{H}\hat{\mathbf{u}}^t} \right), \quad (9)$$

where $\hat{\mathbf{u}}^t$ is the estimate at iteration t , \circ represents the Hadamard product and the quotient is element-wise. The complete derivation is included in Section B.

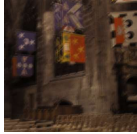
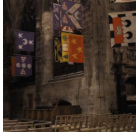
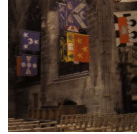


Original	End-to-end				Model-based			
	D-GAN2 [15]	SRN [31]	RealBlur [26]	MPRNet [37]	Whyte [34]	Sun [29]	Gong [5]	Ours
								
27.58 dB	30.88 dB	31.51 dB	33.06 dB	33.37 dB	34.00 dB	27.78 dB	27.45 dB	35.19 dB
								
27.53 dB	31.32 dB	32.21 dB	33.11 dB	26.54 dB	36.25 dB	27.94 dB	27.55 dB	34.93 dB

Figure 6: **Qualitative comparison of different deblurring methods in Köhler Dataset [13] of real blurred images.**

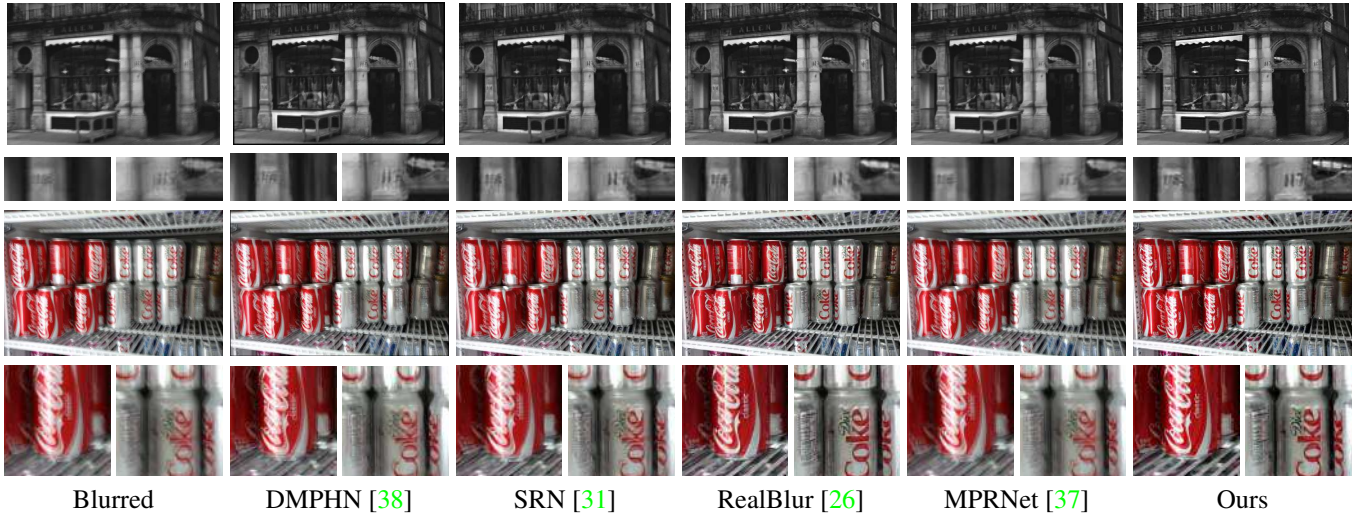


Figure 7: **Deblurring examples on real blurry images from Lai dataset [16].** Our model-based approach is competitive with state-of-the-art data-driven methods. Best appreciated in electronic format.

Non-uniform Richardson-Lucy. To apply the RL update with the proposed non-uniform model, it is necessary to compute the matrix-vector products with matrices \mathbf{H} and \mathbf{H}^T in (9). For this, we benefit from the low-rank decomposition of the kernel field. Denoting $\mathbf{H} = \sum_b \mathbf{M}_b \mathbf{K}_b$, where \mathbf{M}_b is a diagonal matrix containing the per-pixel mixing coefficient of the b^{th} basis kernel and \mathbf{K}_b is a Toeplitz matrix

of the convolution with the basis element \mathbf{k}_b ,

$$\mathbf{H}\mathbf{x} = \sum_{b=1}^B \mathbf{M}_b \mathbf{K}_b \mathbf{x}, \quad \mathbf{H}^T \mathbf{x} = \sum_{b=1}^B \mathbf{K}_b^T \mathbf{M}_b \mathbf{x}. \quad (10)$$

Here we leverage the fact that the product of \mathbf{K}_b or \mathbf{K}_b^T with any vector \mathbf{x} can be computed efficiently via the DFT. A similar procedure was proposed recently in a concurrent

work [7], in the context of modeling nonstationary lens blur.

Richardson-Lucy in the Presence of Saturated Pixels. The presence of saturated pixels in the blurry image may be an indicator of information loss during acquisition, and prevents the perfect recovery of the latent saturated pixels. Additionally, this generates a chain reaction that affects the restoration of nearby pixels. Underestimation of latent saturated pixels due to saturated blurry pixels is one of the main causes of ringing artifacts [33].

To prevent the error propagation caused by the loss of information in blurry saturated pixels, Whyte *et al.* [33] proposed to decouple the estimation of “bright” pixels in \mathbf{u} from the latent pixels in reliable regions by separating them, in each RL update, into two different regions \mathcal{S} and \mathcal{U} , respectively. The segmentation is performed in each iteration using a threshold in the current latent image. In this work, we considered a threshold of 0.99. We applied a smoothing to prevent artifacts due to discontinuities between both regions. The latent image in terms of those disjoint sets is written as: $\mathbf{u} = \mathbf{u}_{\mathcal{U}} + \mathbf{u}_{\mathcal{S}}$, where $\mathbf{u}_{\mathcal{U}}$ is the image where pixels in \mathcal{S} are 0 and vice-versa. Following [32], the update rule for unsaturated pixels is

$$\hat{\mathbf{u}}_{\mathcal{U}}^{t+1} = \hat{\mathbf{u}}_{\mathcal{U}}^t \circ \mathbf{H}^T \left(\frac{\mathbf{v} \circ R'(\mathbf{H}\hat{\mathbf{u}}^t) \circ \mathbf{z}}{R(\mathbf{H}\hat{\mathbf{u}}^t)} + \mathbf{1} - R'(\mathbf{H}\hat{\mathbf{u}}^t) \circ \mathbf{z} \right), \quad (11)$$

and for saturated pixels,

$$\hat{\mathbf{u}}_{\mathcal{S}}^{t+1} = \hat{\mathbf{u}}_{\mathcal{S}}^t \circ \mathbf{H}^T \left(\frac{\mathbf{v} \circ R'(\mathbf{H}\hat{\mathbf{u}}^t)}{R(\mathbf{H}\hat{\mathbf{u}}^t)} + \mathbf{1} - R'(\mathbf{H}\hat{\mathbf{u}}^t) \right). \quad (12)$$

Function R' is the derivative of R defined in (4), while \mathbf{z} is a binary mask whose values are 0 for pixels affected by the loss of information and 1 otherwise [32]. Intuitively, the main difference between (11) and (12) is that to prevent the propagation of ringing, in the former only pixels with no loss of information are used to estimate the latent image while in the latter all the data is used. $R'(x) \simeq 1$ for saturated pixels in the blurry images and therefore those pixels are not taken into account to recover the latent image.

Regularized Richardson-Lucy. A natural image *prior* can be easily added to the RL formulation [3, 30]. The *Maximum a Posteriori* (MAP) solution implies modifying (9) as in [3]:

$$\hat{\mathbf{u}}_{unreg}^{t+1} = \hat{\mathbf{u}}_{\mathcal{S}}^{t+1} + \hat{\mathbf{u}}_{\mathcal{U}}^{t+1} \quad (13)$$

$$\hat{\mathbf{u}}^{t+1} = \frac{\hat{\mathbf{u}}_{unreg}^{t+1}}{1 + \nabla_{\mathbf{u}} \mathcal{L}_{prior}(\hat{\mathbf{u}}^t)}. \quad (14)$$

Here we consider a Total Variation regularization $\mathcal{L}_{prior} = \lambda_{TV} \|\nabla \mathbf{u}\|$.

The RL iteration is initialized with the blurry image, and ran until the reblur loss does not improve or until it reaches 30 iterations.

7.2. Experimental Results in Real Images

In this section we evaluate the ability of our deblurring procedure to generalize to real motion blurred photographs. We compare the deblurring results on the standard datasets of real blurred images: Köhler [13], Lai [16] and RealBlur [26].

Method	Köhler	
	PSNR	SSIM
DeblurGAN [14]	26.05	0.75
GoPro K=2 [20]	(26.02)	(0.81)
SRN [31]	27.18	0.79
DMPHN 1-2-4 [38]	25.69	0.75
DeblurGANv2 Inc. [15]	27.25	0.79
DeblurGANv2 Mob. [15]	25.88	0.74
SRN ¹ [31]	(26.57)	(0.80)
SRN ² [31]	27.85	0.81
MPRNet [37]	26.57	0.78
Sun <i>et. al</i> [29]	(25.22)	(0.77) ¹
Gong [5]	25.23	0.74
Ours	28.39	0.82

Table 1: **Quantitative comparison for image deblurring (PSNR/SSIM) in a real dataset.** When possible, we reproduced the results using available code, otherwise parenthesis are used and values are taken from their own paper. ¹ is trained with RealBlurJ and ² with GoPro, BSD, RealBlurJ as in [26].

Köhler Dataset [13]. Quantitative and qualitative results are presented in Table 1 and Figure 6. Our method compares favorably to state-of-the-art end-to-end DL methods (gain between 0,54 and 2,51 dB), that fail to generalize from the synthetic dataset they were trained on to real blurred images (gain of 3.17 and 3.16 dB).

Lai Dataset [16] is a standard benchmark of real blurry images with no corresponding ground truth, allowing only visual comparison, which we show in Figure 7, 8 and in the Appendix. Note that our model-based method is competitive with state-of-the-art end-to-end approaches, outperforming most of them. Figure 8 shows the capacity of our model to deal with real scenes with saturated regions in low-light conditions. Note that most of the compared methods fall short in this case.

RealBlur Dataset [26]. We evaluate our method following the benchmark presented in [26]. The comparison



Figure 8: **Deblurring examples on real images with saturated regions from Lai dataset [16].** While most of the methods struggle to handle saturated regions, our method is able to recover them. Corresponding kernel maps can be found in the Figure 12. Best viewed in electronic format.

Method	RealBlur-R		RealBlur-J	
	PSNR	SSIM	PSNR	SSIM
Hu et al. [9]	33.67	0.916	26.41	0.803
Pan et al. [24]	34.01	0.916	27.22	0.790
Xu et al. [36]	34.46	0.937	27.14	0.830
Nah et al. [20]	32.51	0.841	27.87	0.827
DeblurGAN [14]	33.79	0.903	27.97	0.834
DeblurGAN-v2 [15]	35.26	0.944	28.70	0.866
Zhang et al. [39]	35.48	0.947	27.80	0.847
SRN [31]	35.66	0.947	28.56	0.867
DMPHN [38]	35.70	0.948	28.42	0.860
MPRNet [37]	35.99	0.952	28.70	0.873
Ours	36.17	0.946	28.95	0.865

Table 2: **Cross-dataset quantitative evaluation.** The first three methods are classical variational methods. DL-based methods were trained on synthetic datasets and evaluated on real blurry images. The PSNR/SSIM scores for other methods are taken from the RealBlur benchmark [26, 37].

to state-of-the-art end-to-end DL-based and classical variational methods is shown in Table 2. All the learning-based compared methods were trained in the GoPro dataset [20] and ours in our dataset presented in Section 4. All methods are evaluated in RealBlur [26]. This cross-dataset evaluation allows to validate the generalization of the deblurring performance and to avoid evaluating dataset-specific mappings. Our method outperforms previous methods in terms of PSNR. We conjecture that this is due to the heavy overfitting of end-to-end DL methods to specifics of the training

databases. Qualitative comparisons on representative examples can be found in Section A.4.

7.3. Blurring to Deblur

Chen *et al.* [1] proposed to impose cycle-consistency to deblurring models using a forward model estimated from consecutive frames in a video. The motivation was to prevent a deblurring conditional GAN [14] from hallucinating image content. In the same spirit, and to validate our estimated kernels, we fine-tuned a pre-trained DeblurGAN [14] network, using our estimated kernels for imposing the forward-model consistency. Results shown in Table 3 prove that our fine-tuning is useful to improve a DeblurGAN model, slightly outperforming [1]. Also we note that different to [1], our method only requires a single frame instead of a video.

Method	Gopro	
	PSNR	SSIM
DeblurGAN	27.25	0.81
DeblurGAN (resume training)	27.57	0.83
DeblurGAN + Reblur2Deblur[1]	28.03 ¹	0.90¹
DeblurGAN + ours	28.08	0.85

Table 3: **Blurring to deblur.** Comparison between Reblur2Deblur [1] and the proposed method in GoPro [20]. The incorporation of a reblur loss in training produces better results than just resuming training. ¹ No code available.

8. Conclusions

We introduced a novel method that predicts a dense field of motion blur kernels, via an efficient image-dependent decomposition. This results in a compact, non-parametric definition of the non-uniform motion field. Extensive experimental results validate the estimated kernels and show that when combined with a non-blind variational deblurring algorithm, the method’s generalization to real blurry images is superior to that of state-of-the-art end-to-end deep learning methods. This work contributes to bridging a longstanding gap between model-based and data-driven methods for motion blur removal.

Acknowledgements

GC was supported partially by Agencia Nacional de Investigación e Innovación (ANII, Uruguay) grant POS_FCE_2018_1_1007783 and PV by the MICINN/FEDER UE project under Grant PGC2018-098625-B-I0; H2020-MSCA-RISE-2017 under Grant 777826 NoMADS and Spanish Ministry of Economy and Competitiveness under the Maria de Maetzu Units of Excellence Programme (MDM-2015-0502). The experiments presented in this paper were carried out using ClusterUY (site: <https://cluster.uy>) and GPUs donated by NVIDIA Corporation. We also thank Juan F. Montesinos for his help during the experimental phase.

References

- [1] H. Chen, J. Gu, O. Gallo, M. Liu, A. Veeraraghavan, and J. Kautz. Reblur2deblur: Deblurring videos via self-supervised learning. In *2018 IEEE International Conference on Computational Photography (ICCP)*, pages 1–9, Los Alamitos, CA, USA, may 2018. IEEE Computer Society. 8
- [2] Mauricio Delbracio and Guillermo Sapiro. Removing camera shake via weighted fourier burst accumulation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(1):234–778, 2002. 3, 11
- [3] Nicolas Dey, Laure Blanc-Féraud, Christophe Zimmer, Pascal Roux, Zvi Kam, Jean-Christophe Olivo-Marin, and Josiane Zerubia. *3D microscopy deconvolution using Richardson-Lucy algorithm with total variation regularization*. PhD thesis, INRIA, 2004. 7
- [4] Fabien Gavant, Laurent Alacoque, Antoine Dupret, and Dominique David. A physiological camera shake model for image stabilization systems. In *SENSORS, 2011 IEEE*, pages 1461–1464, 2014. 1, 3, 11
- [5] Dong Gong, Jie Yang, Lingqiao Liu, Yanning Zhang, Ian Reid, Chunhua Shen, Anton Van Den Hengel, and Qinfeng Shi. From motion blur to motion flow: a deep learning solution for removing heterogeneous motion blur. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2319–2328, 2017. 1, 2, 3, 4, 5, 6, 7, 11, 15, 16, 17, 18
- [6] Ankit Gupta, Neel Joshi, C. Lawrence Zitnick, Michael Cohen, and Brian Curless. Single image deblurring using motion density functions. In *Proceedings of the 11th European Conference on Computer Vision: Part I, ECCV’10*, page 171–184, Berlin, Heidelberg, 2010. Springer-Verlag. 1, 2, 3
- [7] Moonsung Gwak and Seungjoon Yang. Modeling nonstationary lens blur using eigen blur kernels for restoration. *Opt. Express*, 28(26):39501–39523, Dec 2020. 7
- [8] M. Hirsch, C. J. Schuler, S. Harmeling, and B. Schölkopf. Fast removal of non-uniform camera shake. In *2011 International Conference on Computer Vision*, pages 463–470, 2011. 1, 2, 3
- [9] Zhe Hu, Sunghyun Cho, Jue Wang, and Ming-Hsuan Yang. Deblurring low-light images with light streaks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3382–3389, 2014. 8
- [10] 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. 3
- [11] T. H. Kim, B. Ahn, and K. M. Lee. Dynamic scene deblurring. In *2013 IEEE International Conference on Computer Vision*, pages 3160–3167, 2013. 2
- [12] T. H. Kim and K. M. Lee. Segmentation-free dynamic scene deblurring. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2766–2773, 2014. 1
- [13] Rolf Köhler, Michael Hirsch, Betty Mohler, Bernhard Schölkopf, and Stefan Harmeling. Recording and playback of camera shake: Benchmarking blind deconvolution with a real-world database. In *European conference on computer vision*, pages 27–40. Springer, 2012. 6, 7, 17
- [14] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8183–8192, 2018. 2, 7, 8
- [15] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8878–8887, 2019. 2, 6, 7, 8, 17
- [16] Wei-Sheng Lai, Jia-Bin Huang, Zhe Hu, Narendra Ahuja, and Ming-Hsuan Yang. A comparative study for single image blind deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recog.*, pages 1701–1709, 2016. 4, 6, 7, 8, 14, 15, 16
- [17] B. L. Lucy. An iterative technique for the rectification of observed distributions. *Astronomical Journal*, 1974. 2, 19
- [18] 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. 3
- [19] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *The IEEE Conference*

- on *Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019. 2, 4, 5
- [20] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2, 4, 5, 7, 8
- [21] 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. 3
- [22] 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. 3
- [23] Jin-shan Pan, Zhe Hu, Zhixun Su, Hsin-Ying Lee, and Ming-Hsuan Yang. Soft-segmentation guided object motion deblurring. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 459–468. IEEE Computer Society, 2016. 2
- [24] Jinshan Pan, Deqing Sun, Hanspeter Pfister, and Ming-Hsuan Yang. Blind image deblurring using dark channel prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1628–1636, 2016. 8
- [25] William Hadley Richardson. Bayesian-based iterative method of image restoration*. *J. Opt. Soc. Am.*, 62(1):55–59, Jan 1972. 2, 19
- [26] Jaesung Rim, Haeyun Lee, Jucheol Won, and Sunghyun Cho. Real-world blur dataset for learning and benchmarking deblurring algorithms. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2, 6, 7, 8, 11, 14, 17, 18
- [27] Shengyang Dai and Ying Wu. Motion from blur. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 1
- [28] Shuochen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. Deep video deblurring for hand-held cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1279–1288, 2017. 2
- [29] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal. In *Proceedings of the IEEE Conference on Computer Vis.*, 2015. 1, 2, 4, 5, 6, 7, 11, 15, 16, 17, 18
- [30] Y. Tai, P. Tan, and M. S. Brown. Richardson-lucy deblurring for scenes under a projective motion path. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1603–1618, 2011. 2, 5, 7
- [31] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recog. Worksh.*, 2018. 2, 6, 7, 8, 14, 17
- [32] Oliver Whyte. *Removing camera shake blur and unwanted occluders from photographs*. PhD thesis, École normale supérieure de Cachan-ENS Cachan, 2012. 7
- [33] Oliver Whyte, Josef Sivic, and Andrew Zisserman. Deblurring shaken and partially saturated images. *International journal of computer vision*, 110(2):185–201, 2014. 3, 7
- [34] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. c. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 491–498, 2010. 1, 2, 3, 6, 17
- [35] Zhihao Xia, Federico Perazzi, Michaël Gharbi, Kalyan Sunkavalli, and Ayan Chakrabarti. Basis prediction networks for effective burst denoising with large kernels. *arXiv preprint arXiv:1912.04421*, 2019. 3, 4, 11
- [36] Li Xu, Shicheng Zheng, and Jiaya Jia. Unnatural l0 sparse representation for natural image deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1107–1114, 2013. 8
- [37] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *CVPR (to appear)*, 2021. 6, 7, 8, 14, 17
- [38] Hongguang Zhang, Yuchao Dai, Hongdong Li, and Piotr Koniusz. Deep stacked hierarchical multi-patch network for image deblurring. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 200319. 2, 6, 7, 8, 14
- [39] Jiawei Zhang, Jinshan Pan, Jimmy Ren, Yibing Song, Linchao Bao, Rynson WH Lau, and Ming-Hsuan Yang. Dynamic scene deblurring using spatially variant recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2521–2529, 2018. 8
- [40] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3, 11
- [41] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *2011 International Conference on Computer Vision*, pages 479–486. IEEE, 2011. 3

A. Appendix

A.1. Architecture Details

Our network is built upon a Kernel Prediction Network (KPN) proposed by [35]. The network architecture is shown in Figure 3 of the submission. It is composed of one *contractive path* and two expansive paths that yield the kernel basis and mixing coefficients. We call them *kernel head* and *mixing coefficients head*, respectively.

The *contractive path* is composed of five *down-sampling* blocks. Each *down-sampling* block is composed of two convolutions followed by a max-pooling layer. After each *down-sampling* block the spatial size is divided by two. The output is a feature vector, that is fed into both the *kernel head* and *mixing coefficients head*.

The *mixing coefficients head* together with the *contractive path* follow a U-Net architecture with skip connections. After the last convolution, softmax is applied along the B -channels dimension to ensure that the sum of the per-pixel mixing coefficients associated with the kernels adds up to one.

The *kernel head* aims to produce B basis kernels, each of size $K \times K$. In our work $B = 25$ and $K = 33$. The first operation performed is a Global Average Pooling which takes into account that, unlike the *mixing coefficients head*, there is no correspondence between the spatial positions of input and output pixels. Then, the feature vector goes through five *up-sampling* blocks. Each *up-sampling* is composed of a bilinear upsampling followed by three convolutional layers. Additionally, there are skip connections between the second convolution in the *down-sampling* blocks and the second convolution in the *up-sampling* blocks. Finally, two 64-channels convolutional layers followed by a B -channels convolutional layer with softmax are applied to ensure the B kernels of size $K \times K$ are positive and add up to one. Convolutional kernel sizes are 3 with 1-padding except for the first 64-channel convolutions, whose kernel sizes are 2.

Figure 2 shows additional examples of kernels and corresponding mixing coefficients generated with the proposed network. The non-uniform motion blur represented by the set of basis kernels and mixing coefficients allows to re-blur the corresponding sharp image by first convolving it with each basis kernel, and then performing a weighted sum of the results using the mixing coefficients.

A.2. Training Details

We train the proposed network with a combination of the L^2 -norm and L^1 -norm for a total number of 1200 epochs in two steps. In the first 300 epochs, the network is trained using an L^2 -norm in the *kernels loss*, continued by 900 epochs switching to L^1 -norm. We minimize our objective loss using Adam optimizer with standard hyperparameters. We start with a learning rate equal to $1e-4$ and halve it every

150 epochs.

Table 4 compares the performance obtained for different numbers of basis elements and different training schemes.

Method	B=15	B=25	B=40
L2 300 epochs	28.52	28.64	28.56
L2 450 epochs	28.64	28.70	28.86
L2 300 epochs+L1 150 epochs	28.95	28.85	28.93

Table 4: Comparison of restoration performance for models trained with different values of B . Models were evaluated in the RealBlurJ test dataset using the same restoration algorithm. Results shown are PSNRs values.

A.3. Dataset Generation

To generate the training set for the kernel estimation network, we propose a method that generates non-uniform motion blurred images, based on the training set of the ADE20K semantic segmentation dataset [40]. We choose a subset of 5888 images containing at least one segmented person or car having a minimum size of 400 pixels. To blur the images, we use a dataset of 500,000 kernels with a maximum exposure time of one second. The dataset of kernels was generated by a camera-shake kernel generator [4, 2] based on physiological hand tremor data. In this way we obtain for each image a tuple $(\mathbf{u}^{GT}, \mathbf{v}^{GT}, \{\mathbf{k}\}^{GT}, \{\mathbf{m}\}^{GT})$ composed by the ground truth sharp image, the blurry image, the pairs of ground truth blur kernels and masks, respectively. Notice that the kernels $\{\mathbf{k}\}^{GT}$ are not the kernel basis as computed by our network but the resulting ground truth kernels present in the image. A pseudo-code of the blurry synthetic dataset generation is presented Algorithm 1.

A.4. Kernel Generation and Image Restoration on Real Images

Figure 11 compares our results to those obtained with state-of-the-art deep learning-based methods on Lai’s dataset. A comparison of our results on the same database with existing non-uniform motion blur estimation methods is presented in Figures ?? and 13. Figure 14 shows additional deblurring examples from Köhler’s dataset.

Additional examples of generated kernels together with the estimated deblurred images on RealBlur [26] are shown in Figure 15. On the RealBlur examples, results are compared to other kernel estimation methods (Gong *et al.* [5] and Sun *et al.* [29]). Notice that, contrarily to these approaches, the kernels estimated by our method show almost no correlation with the image structure. Moreover, our approach is capable of retrieving better estimates in regions exhibiting low contrast, which are often present in motion blurred images suffering from limited exposure time.

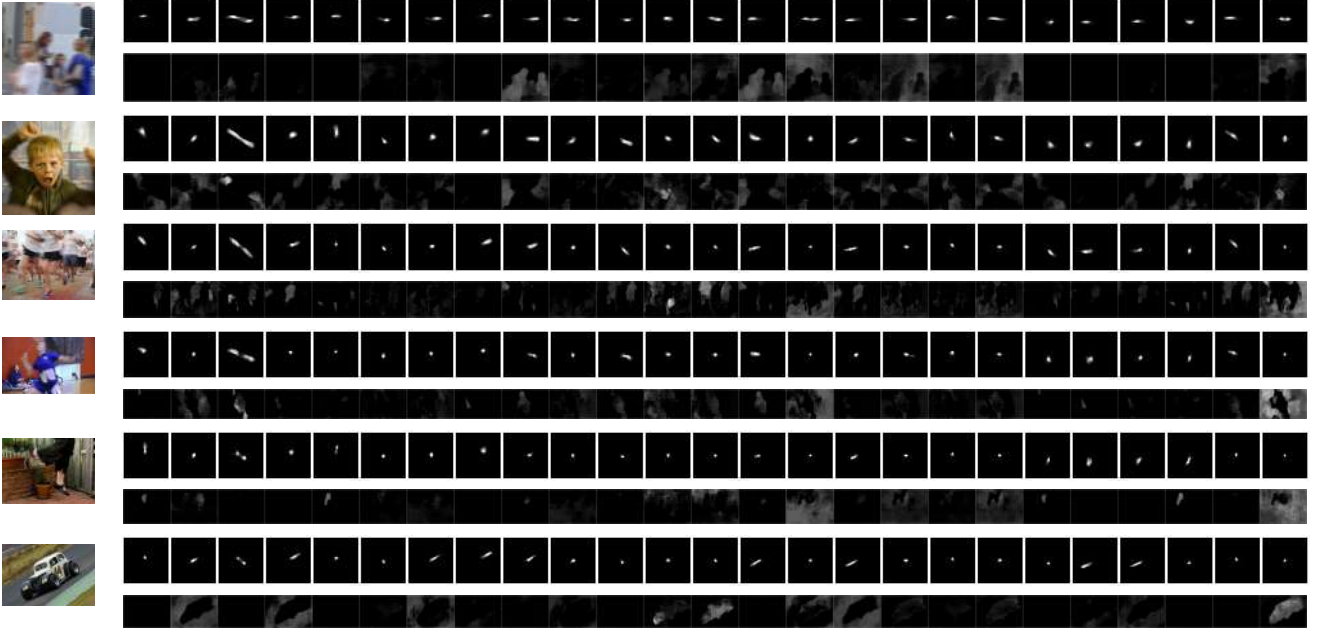


Figure 9: **Examples of generated kernel basis $\{\mathbf{k}^b\}$ and corresponding mixing coefficients $\{\mathbf{m}^b\}$** predicted from the blurry images shown on the left. The adaptation to the input is more notorious for the elements that have significant weights.

Algorithm 1 Synthetic Image Generation.

Input

$\mathbf{u}, \{\mathbf{k}\}$ \triangleright Sharp Image and dataset of kernels

procedure BLURIMAGE($\mathbf{u}, \{\mathbf{k}\}$)

$\mathbf{k}_u.append(Random(\{\mathbf{k}\}))$ \triangleright Background kernel

$\mathbf{m}_u.append(ones(size(\mathbf{u})))$ \triangleright Background mask

$\mathbf{v} = zeros(size(\mathbf{u}))$ \triangleright Initialize blurry image

for $\mathbf{m} \in SegmentedObjectMasks(\mathbf{u})$ **do**

$\mathbf{k} = Random(\{\mathbf{k}\})$ \triangleright Object kernel

$\mathbf{m}_u[0] = \mathbf{m}_u[0] - \mathbf{m}$ \triangleright Update background mask

$\mathbf{k}_u.append(\mathbf{k})$

$\mathbf{m}_u.append(\mathbf{m})$

end for

for $\mathbf{k}, \mathbf{m} \in \mathbf{k}_u, \mathbf{m}_u$ **do**

$\mathbf{m} = \mathbf{m} * \mathbf{k}$ \triangleright Smooth mask

$\mathbf{v} = \mathbf{v} + \mathbf{m}(\mathbf{k} * \mathbf{u})$ \triangleright Output image update

end for

end procedure

return $\mathbf{v}, \mathbf{k}_u, \mathbf{m}_u$ \triangleright Blurred Image, kernels and masks



Figure 10: Examples of synthetic blurred images generated by the proposed procedure. From left to right: camera-shake motion kernels from physiological tremor model; image from the ADE20K segmentation dataset; corresponding ADE20K segmentation masks; resulting motion blurred image, obtained by convolving each region in the image with the corresponding kernel.

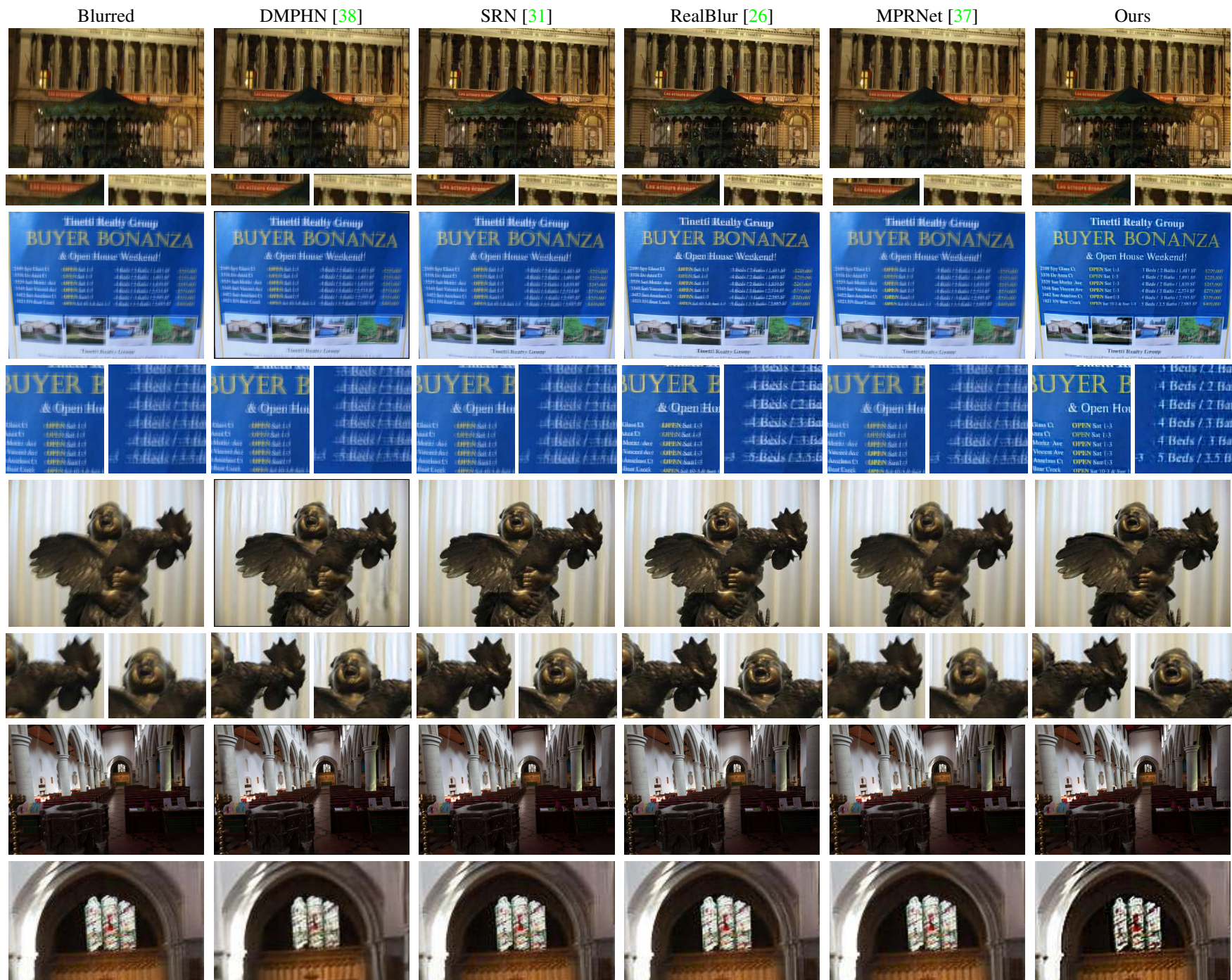


Figure 11: Deblurring examples on real blurry images from Lai’s dataset [16].

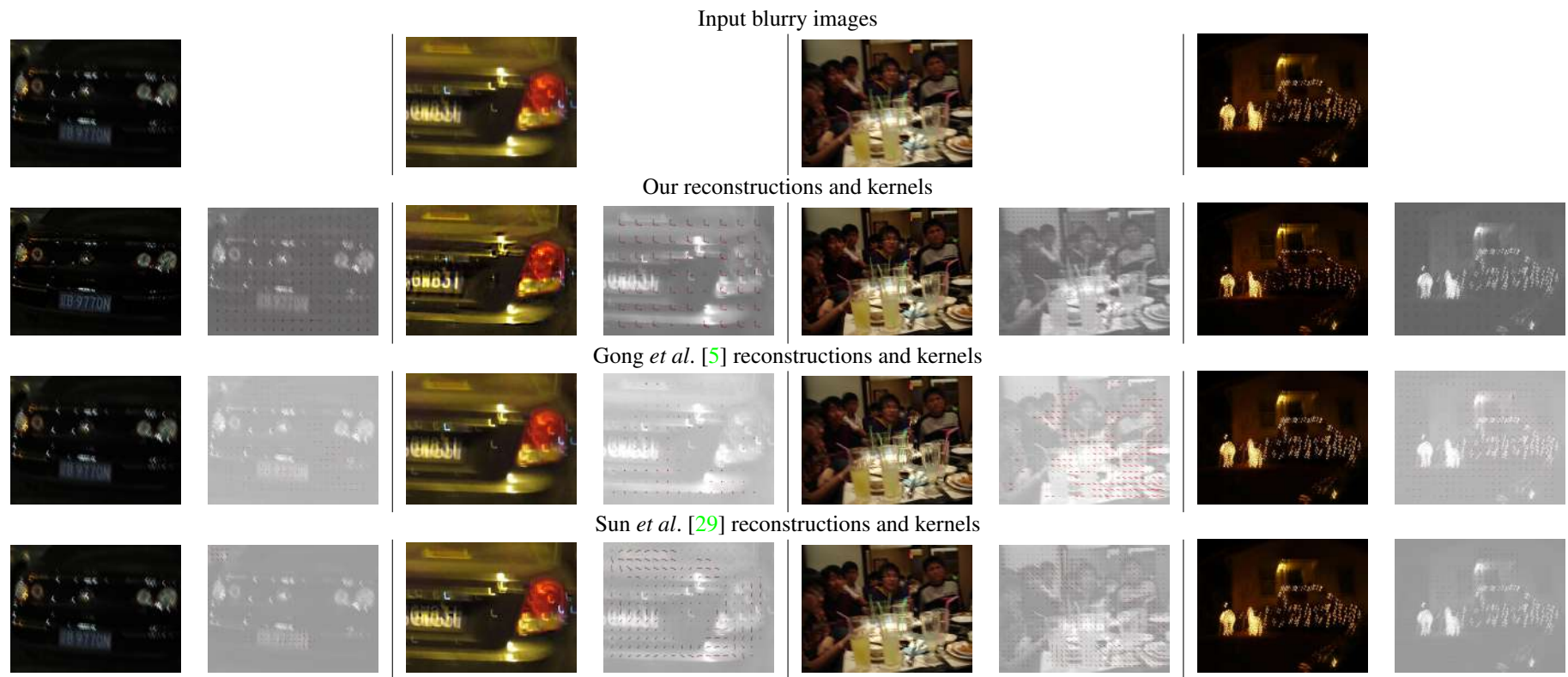


Figure 12: Examples of kernels predicted by our method and corresponding deblurred images on the Lai dataset [16] at half resolution. Comparison with [5] and [29]. Note that these approaches show a significant correlation with the image structure, and are more prone to fail at capturing the motion structure in low contrasted regions.

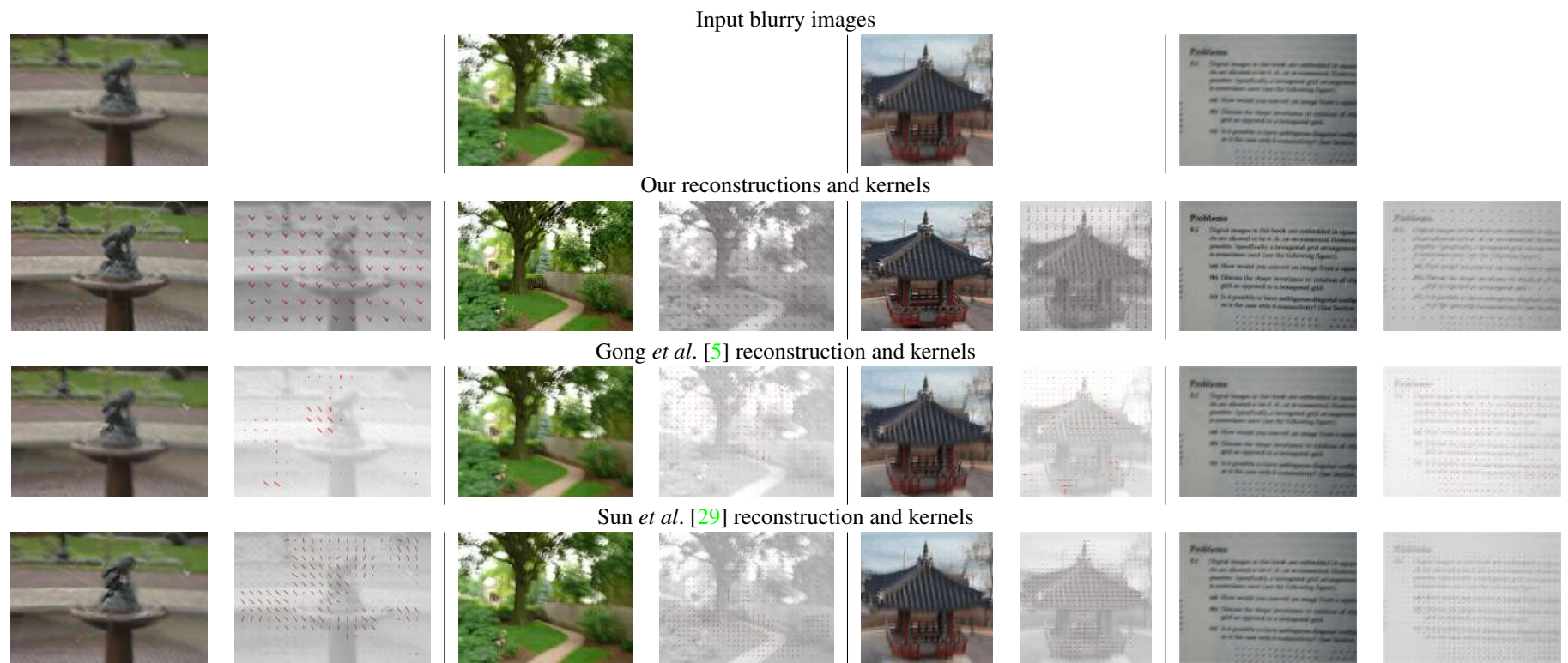


Figure 13: Non-uniform kernel estimation and reconstruction examples for images of Lai’s dataset [16] ran at half resolution.




























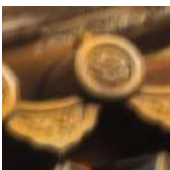








Original	End-to-end					Model-based			
	D-GAN2 [15]	SRN [31]	RealBlur [26]	MPRNet [37]	Whyte [34]	Sun [29]	Gong [5]	Ours	
									
									
21.89 dB	26.54 dB	24.73 dB	28.11 dB	24.08 dB	27.06 dB	22.31 dB	21.87 dB	28.20 dB	
									
									
23.08 dB	25.94 dB	25.14 dB	28.04 dB	24.78 dB	29.57 dB	23.41 dB	23.08 dB	29.00 dB	

Figure 14: Qualitative comparison of different deblurring methods on Köhler’s Dataset [13].

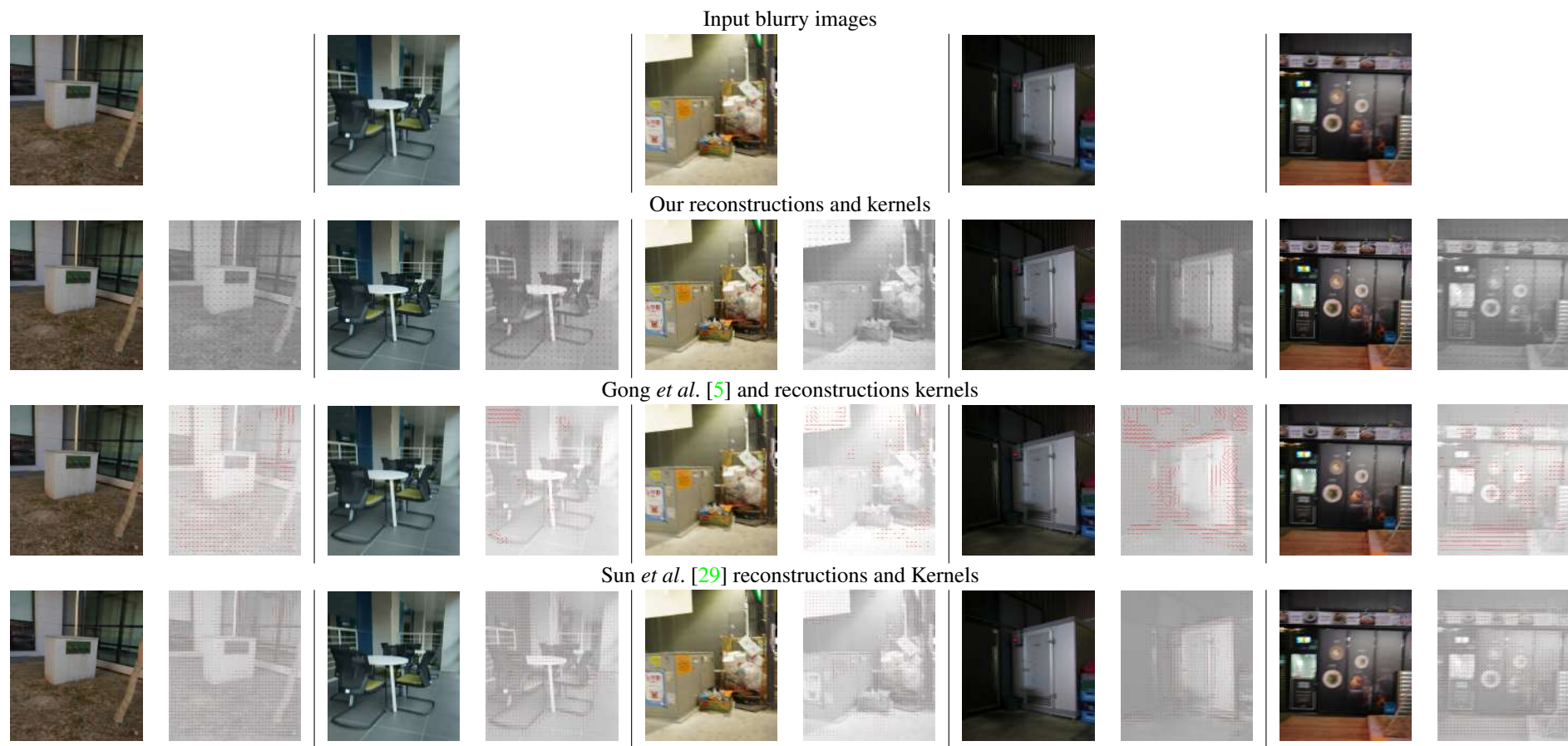


Figure 15: Examples of kernels predicted by our method and corresponding deblurred images on the RealBlur dataset [26], and comparison with [5] and [29]. Note that these approaches show a significant correlation with the image structure, and are more prone to fail at capturing the motion structure in low contrasted regions.

B. Richardson-Lucy derivation

For the sake of completeness, in this section we summarize the derivation of the Richardson-Lucy iteration [?]. Richardson-Lucy [25, 17], algorithm recovers the latent sharp image as the *maximum-likelihood* estimate under a *Poisson noise* model. The likelihood of the blurry image \mathbf{v} given the latent image \mathbf{u} is given by

$$p(\mathbf{u}|\mathbf{v}) = \prod_i \frac{\hat{v}_i^{v_i} \exp^{-\hat{v}_i}}{v_i!}, \quad (15)$$

where

$$\hat{v}_i = \sum_j \langle \bar{\mathbf{u}}_{i,j}, \mathbf{k}_{i,j} \rangle = \sum_j H_{ij} u_j. \quad (16)$$

Here, \mathbf{H} denotes the *degradation* matrix of size $N_v \times N_v$, where N_v is the number of pixels in the image. Each column of \mathbf{H} contains the *Point Spread Function* for the corresponding pixel in the latent image \mathbf{u} . Maximizing (15) is equivalent to minimizing the negative log-likelihood

$$\mathcal{L}_{pois} = \sum_i \hat{v}_i - v_i \log \hat{v}_i + \log(v_i!). \quad (17)$$

The cost function is optimized over the latent image \mathbf{u} , under the constraint $u_j \geq 0$, for all pixel j . Therefore, the optimum must satisfy the Karush-Kuhn-Tucker conditions for every pixel:

$$\begin{cases} u_j \frac{\partial}{\partial u_j} \mathcal{L}_{pois} = 0 & u_j > 0, \\ \frac{\partial}{\partial u_j} \mathcal{L}_{pois} \geq 0 & u_j = 0. \end{cases} \quad (18)$$

$$\quad (19)$$

The Richardson-Lucy algorithm builds up on (18). From this,

$$u_j \sum_i \frac{\partial}{\partial u_j} (\hat{v}_i - v_i \log \hat{v}_i + \log(v_i!)) = 0 \quad (20)$$

$$u_j \sum_i \left(\frac{\partial \hat{v}_i}{\partial u_j} - \frac{v_i}{\hat{v}_i} \frac{\partial \hat{v}_i}{\partial u_j} \right) = 0 \quad (21)$$

$$u_j \sum_i H_{ij} = u_j \sum_i H_{ij} \frac{v_i}{\hat{v}_i}. \quad (22)$$

Since the blur is conservative, the sum of each column of \mathbf{H} is equal to one. Hence,

$$u_j = u_j \sum_i H_{ij} \frac{v_i}{\hat{v}_i}. \quad (23)$$

This equality corresponds to a fixed point equation, that can be solved *via* the following update rule:

$$\hat{u}_j^{t+1} = \hat{u}_j^t \sum_i H_{ij} \frac{v_i}{(\mathbf{H}\hat{\mathbf{u}})_i}. \quad (24)$$

In matrix-vector form, this writes

$$\hat{\mathbf{u}}^{t+1} = \hat{\mathbf{u}}^t \circ \mathbf{H}^T \left(\frac{\mathbf{v}}{\mathbf{H}\hat{\mathbf{u}}^t} \right), \quad (25)$$

where the \circ denotes the Hadamard product and the fraction represents the element-wise division of two vectors. Note that choosing an initial condition $\hat{\mathbf{u}}^0 \geq 0$ ensures that $\hat{\mathbf{u}}^t \geq 0$ for all $t > 0$.