Learning Blind Motion Deblurring

Patrick Wieschollek[1,2] Michael Hirsch[2] Bernhard Schölkopf[2]
Hendrik P.A. Lensch[1]

[1] University of Tübingen
[2] Max Planck Institute for Intelligent Systems, Tübingen

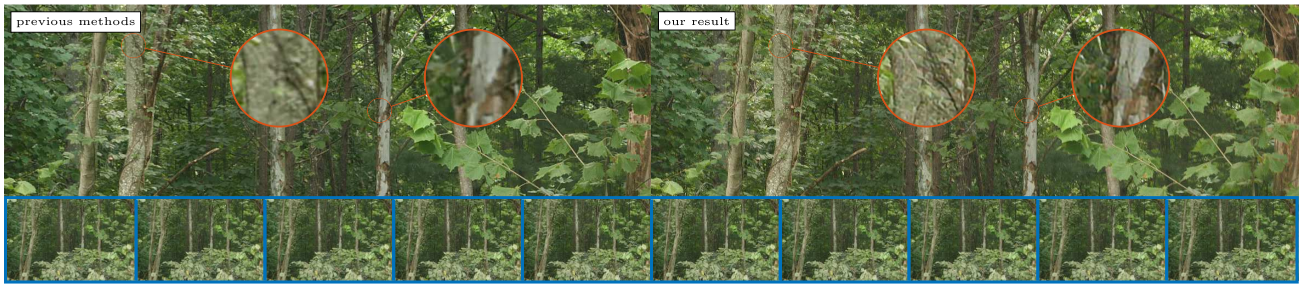

Figure 1. From a sequence of blurry inputs (lower row) our learning-based approach for blind burst-deblurring reconstructs fine details, which are not recovered by recent state-of-the-art methods like FBA [6]. Both methods feature similar run-time.

Abstract

As handheld video cameras are now commonplace and available in every smartphone, images and videos can be recorded almost everywhere at anytime. However, taking a quick shot frequently yields a blurry result due to unwanted camera shake during recording or moving objects in the scene. Removing these artifacts from the blurry recordings is a highly ill-posed problem as neither the sharp image nor the motion blur kernel is known. Propagating information between multiple consecutive blurry observations can help restore the desired sharp image or video. In this work, we propose an efficient approach to produce a significant amount of realistic training data and introduce a novel recurrent network architecture to deblur frames taking temporal information into account, which can efficiently handle arbitrary spatial and temporal input sizes.

1. Introduction

Videos captured by handheld devices usually contain motion blur artifacts caused by a combination of camera shake (ego-motion) and dynamic scene content (object motion). With a fixed exposure time any movement during recording causes the sensor to observe an averaged signal from different points in the scene. A reconstruction of the sharp frame from a blurry observation is a highly ill-posed problem, denoted as blind or non-blind deconvolution depending on whether camera-shake information is known or not.

In video and image burst deblurring the reconstruction process for a single frame can make use of additional data from neighboring frames. However, the problem is still challenging as each frame might encounter a different camera shake and the frames might not be aligned.

For deconvolution of a static scene neural networks have been successfully applied using single frame [2, 21, 22] and multi-frame deblurring [30, 33, 5].

All recent network architectures for multi-frame and video deblurring [30, 24, 17, 2] require the input to match a fixed temporal *and* spatial size. Handling arbitrary spatial dimensions is theoretically possible by fully convolutional networks as done in [24], but they rely on a sliding window approach during inference due to limited memory on the GPU. For these approaches, the reconstruction of one frame is not possible by aggregating the information of longer sequences than the network was trained for.

In contrast, our approach is a deblurring system that can deal with arbitrary lengths of sequences while featuring a fully convolutional network that can process full resolution video frames at once. Due to its small memory footprint it removes the need for sliding window approaches during inference, thus drastically accelerating the deblurring process. For processing arbitrary sequences we rely on a recurrent scheme. While convolutional LSTMs [18] offer a straight-

forward way to replace spatial convolutions in conventional architectures by recurrent units, we found them challenging and slow to train. Besides vanishing gradients effects they require a bag of tricks like carefully tuned gradient clipping parameters and a special variant of batch normalization. In order to circumvent these problems, we introduce a new recurrent encoder-decoder network. In the network we incorporate spatial residual connections and introduce novel temporal feature transfer between subsequent iterations.
Besides the network architecture, we further create a novel training set for video deblurring as the success of data-driven approaches heavily depends on the amount and quality of available realistic training examples. As acquiring realistic ground-truth data is time-consuming, we successfully generate synthetic training data with literally no acquisition cost and demonstrate improved results and run time on various benchmark sets as for example demonstrated in Figure 1.

2. Related Work

The problem of image deblurring can be formulated as a non-blind or a blind deconvolution version, depending on whether information about the blur is available or not. Blind image deblurring (BD) is quite common in real-world applications and has seen considerable progress in the last decade. A comprehensive review is provided in the recent overview article by Wang and Tao [29]. Traditional state-of-the-art methods such as Sun *et al*. [26] or Michaeli and Irani [16] use carefully chosen patch-based priors for sharp image prediction. Data-driven methods based on neural networks have demonstrated success in non-blind restoration tasks [21, 31, 20] as well as for the more challenging task of BD where the blur kernel is unknown [22, 25, 2, 10, 27]. Removing the blur from moving objects has been recently addressed in [17].
To alleviate the ill-posedness of the problem [7], one might take multiple observations into account. Hereby, observations of a static scene, each of which is differently blurred, serve as inputs [33, 3, 1, 23, 35, 9]. To incorporate video properties such as temporal consistency the methods of [32, 34, 13, 12] use powerful and flexible generative models to explicitly estimate the unknown blur along with predicting the latent sharp image. However, this comes at the price of higher computation cost, which typically requires tens of minutes for the restoration process.
To accomplish faster processing times Delbracio and Sapiro [6] have presented a clever way to average a sequence of input frames based on Lucky Imaging methods. They propose to compute a weighted combination of all aligned input frames in the Fourier domain which favors stable Fourier coefficients in the burst containing sharp information. This yields much faster processing times and removes the requirement to compute the blur kernel explicitly.
Quite recently, Wieschollek *et al*. in [30] introduce an end-to-end trainable neural network architecture for multi-frame deblurring. Their approach directly computes a sharp image by processing the input burst in a patch-wise fashion, yielding state-of-the-art results. It has been shown, that this even enables treating spatially varying blur. The related task of deblurring of videos has been approached by Su *et al*. [24]. Their approach uses the U-Net architecture [19] with skip connection to directly regress the sharp image from an input burst. Their fully convolutional neural network learns an average of multiple inputs with reasonable performance. Unfortunately, both learning methods [30, 24] require to fix the temporal input size at training time and they are limited to a patch-based inference by the network layout [30] and memory constraints [24].

3. Method

Overview. In our approach a fully-convolutional neural network deblurs a frame $\mathcal{I}$ using information from previous frames $\mathcal{I}_{-1}, \mathcal{I}_{-2}, \ldots$ in an iterative, recurrent fashion. Incorporating a previous (blurry) observation improves the current prediction for $\mathcal{I}$ step by step. We will refer to these steps as *deblur steps*. Hence, the complete *recurrent deblur network* (RDN) consists of several deblur blocks (DB). We use weight-sharing between these to reduce the total amount of used parameters and introduce novel temporal skip connections between these deblur blocks to propagate latent features between the individual temporal steps. To effectively update the network parameters we unroll these steps during training. At inference time, the inputs can have arbitrary spatial dimensions as long as the processing of a minimum of two frames fits on the GPU. Moreover, the recurrent structure allows us to include an arbitrary number of frames helping to improve the output with each iteration. Hence, there is no need for padding burst sequences to match the network architecture as e.g. in [30].

3.1. Generating realistic ground-truth data

Training a neural network to predict a sharp frame of a blurry input requires realistic training data featuring these two aligned versions for each video frame: a blurry version serving as the input and an associated sharp version serving as ground-truth. Obtaining this data is challenging as any recorded sequence might suffer from the described blur effects itself. Recent work [24, 17] have built a training data set by recording videos captured at 240fps with a GoPro Hero camera to minimize the blur in the ground-truth. Frames from these high-fps videos are then processed and averaged to produce plausible motion blur synthetically. While they made significant effort to capture a broad range of different situations, this process is limited in the number of recorded samples, in the variety of scenes and in the used recording devices. For fast moving objects artifacts

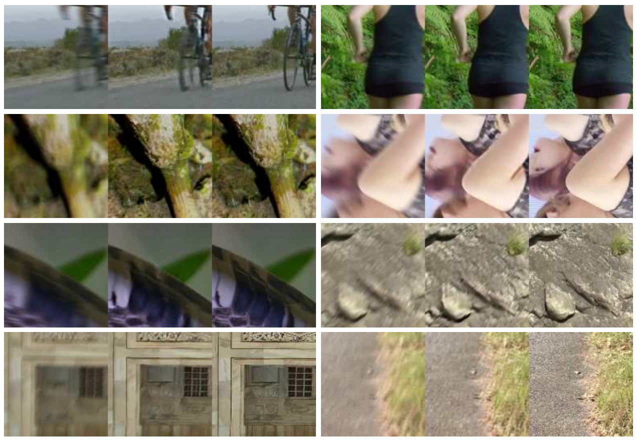

Figure 2. Snapshot of the training process. Each triplet shows the input with synthetic blur (left), the current network prediction (middle) and the associated ground-truth (right). All images are best viewed at higher resolution in the electronic version.

are likely to arise due to the finite framerate. We also tested this method for generating training data with a GoPro Hero camera but found it hard to produce a large enough dataset of sharp ground-truth videos of high quality. Rather than acquiring training data manually we propose to acquire and filter data from online media.

Training data. As people love to share and rate multimedia content, each year millions of video clips are uploaded to online platforms like YouTube. The video content ranges from short clips to professional videos of up to 8k resolution. From this source, we have collected videos with 4k-8k resolution and a frame rate of 60fps or 30fps. The video content ranges from movie trailers, sports events, advertisements to videos on everyday life. To remove compression artifacts and to obtain slightly sharper ground-truth we resized all collected videos by factor $1/4$ respectively $1/8$, finally obtaining full-HD resolution.

Consider such a video with frames $(f_t)_{t=1,2,\dots,T}$. For each frame pair (f_t, f_{t+1}) at time t we compute n additional *synthetical* subframes between the original frames $\mathbf{f_t}, \mathbf{f_{t+1}}$ resulting in a high frame rate video

$$(\dots, f_{t-1}^{(n-1)}, f_{t-1}^{(n)}, \mathbf{f_t}, f_t^{(1)}, f_t^{(2)}, \dots, f_t^{(n-1)}, f_t^{(n)}, \mathbf{f_{t+1}}).$$

All subframes are computed by blending between the neighboring original frames f_t and f_{t+1} warping both frames using the optical flow in both directions $w_{f_t \to f_{t+1}}$ and $w_{f_{t+1} \to f_t}$. Given both flow fields, we can generate an arbitrary number of subframes. For practical purposes, we set $n = 40$, thus implying an effective framerate of more than 1000fps without suffering from low signal-to-noise ratio (SNR) due to short exposure times.

We want to stress that only parts of videos with reasonably sharp frames serve as ground-truth. For those the estimation of optical flow to approximate motion blur is possible and sufficient. The sub-frames are averaged to generate a plausible blurry version

$$b_t = \frac{1}{1+2L}\left(\mathbf{f_t} + \sum_{\ell=1}^{L} f_{t-1}^{(n-\ell)} + f_t^{(\ell)}\right) \qquad (1)$$

for each sharp frame f_t. We use a mix of 20 and 40 for L to create different levels of motion blur. The entire computation can be done offline on a GPU. For all video parts that passed our sharpness test (5.43 hours in total) we produce a ground-truth video and blurry version both at 30 fps in full-HD. Besides the unlimited amount of training data another major advantage of this method is that it incorporates different capturing devices naturally. Further, the massive amount of available video content allows us to tweak all thresholds and parameters in a conservative way to reject video parts of bad quality (too dark, too static) without affecting the effective size of the training data. Though the recovered optical flow is not perfect we observed an acceptable quality of the synthetically motion blurred dataset. To add variety to the training data we crop random parts from the frames and resize them to 128×128px. Figure 2 shows a few random examples from our training dataset.

3.2. Handling the time dimension

The typical input shape required by CNNs in computer vision tasks is $[B, H, W, C]$ – batch size, height, width and number of channels. However, processing series of images includes a new dimension: time. To apply spatial convolution layers the additional dimension has to be "merged" either into the channel $[B, H, W, C \cdot T]$ or batch dimension $[B \cdot T, H, W, C]$. Methods like [24, 30] stack the time along the channel dimension rendering all information across the entire burst available without further modification. This comes at the price of removing information about the temporal order. Further, the number of input frames needs to be fixed before training, which limits their application. Longer sequences could only be processed with workarounds like padding and sliding window processing. On the other hand, merging the time-dimension into the batch dimension would give flexibility at processing different length of sequences. But the processing of each frame is then entirely decoupled from its adjacent frames – no information is propagated. Architectures using convLSTM [18] or convGRU cells [4] are designed to naturally handle time series but they would require several tricks [15, 28] during training. We tried several architectures based on these recurrent cells but found them hard to train and observed hardly any improvement even after two days of training.

3.3. Network Architecture

Instead of including recurrent layers, we propose to formulate the entire network as a recurrent application of deblur

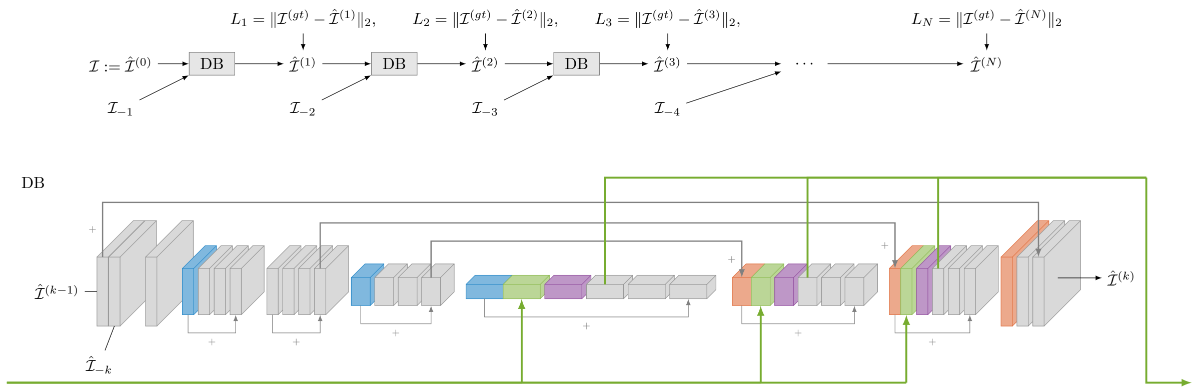
$L_1 = \|\mathcal{I}^{(gt)} - \hat{\mathcal{I}}^{(1)}\|_2,$
$L_2 = \|\mathcal{I}^{(gt)} - \hat{\mathcal{I}}^{(2)}\|_2,$
$L_3 = \|\mathcal{I}^{(gt)} - \hat{\mathcal{I}}^{(3)}\|_2,$
$L_N = \|\mathcal{I}^{(gt)} - \hat{\mathcal{I}}^{(N)}\|_2$
$\mathcal{I} := \hat{\mathcal{I}}^{(0)}$
DB
$\hat{\mathcal{I}}^{(1)}$
$\hat{\mathcal{I}}^{(2)}$
$\hat{\mathcal{I}}^{(3)}$
$\hat{\mathcal{I}}^{(N)}$
$\mathcal{I}_{-1}$
$\mathcal{I}_{-2}$
$\mathcal{I}_{-3}$
$\mathcal{I}_{-4}$
DB
$\hat{\mathcal{I}}^{(k-1)}$
$\hat{\mathcal{I}}_{-k}$
$\hat{\mathcal{I}}^{(k)}$
+

Figure 3. Given the current deblurred version of $\mathcal{I}$ each deblur block DB produces a sharper version of $\mathcal{I}$ using information contributed by another observation $\mathcal{I}_{-k}$. The deblur block follows the design of an encoder-decoder network with several residual blocks with skip-connections. To share learned features between various observations, we propagate some previous features into the current DB (green).

blocks and successively process pairs of inputs (target frame and additional observation), which gives us the flexibility to handle arbitrary sequence lengths and enables information fusion inside the network.

Consider a single deblur step with the current prediction $\mathcal{I}$ of shape $[H, W, C]$ and blurry observation $\mathcal{I}_{-k}$. Inspired by the work of Ronneberger *et al.* [19] and the recent success of residual connections [8] we use an encoder-decoder architecture in each deblur block, see Figure 3. Hereby, the network only consists of convolution and transpose-convolution layers with batchnorm [11]. We applied the ReLU activation to the input of the convolution layers $C_{\cdot,\cdot}$ as proposed in [8].

The first trainable convolution layer expands the 6-channel input (two 128×128px RGB images during training) into 64 channels. In the encoder part, each residual block consists of a down-sampling convolution layer (🟦) followed by three convolution layers (🟪). The down-sampling layer halves the spatial dimension with stride 2 and doubles the effective number of channels $[H, W, C] \to [H/2, W/2, C \cdot 2]$. During the decoding step, the transposed-convolution layer (🟧) inverts the effect of the downsampling $[H, W, C] \to [2 \cdot H, 2 \cdot W, C/2]$. We use a filter size of 3×3 / 4×4 for all convolution/transposed-convolution layers. In the beginning an additional residual block without downsampling accounts for resolving larger blur by providing a larger receptive field.

To speed up the training process, we add skip-connections between the encoding and decoding part. Hereby, we add the extracted features from the encoder to the related decoder part. This enables the network to learn a residual between the blurry input and the sharp ground-truth rather than ultimately generating a sharp image from scratch. Hence, the network is fully-convolutional and therefore allows for arbitrary input sizes. Please refer to Table 1 for more details.

Table 1. Network Specification. Outputs of layers marked with * are concatenated with features from previous deblur blocks except in the first step. This doubles the channel size of the output. The blending layers $B_{\cdot,\cdot}$ are only used after the first deblur step.

layer	filter size	stride	output shape
⬜ $A_{0,1}$	$3 \times 3 \times 64$	2	$^H/_1 \times {}^H/_1 \times 64$
🟦 $C_{1,1}$	$3 \times 3 \times 64$	2	$^H/_2 \times {}^H/_2 \times 64$
⬜ $C_{1,2} - C_{1,4}$	$3 \times 3 \times 64$	1	$^H/_2 \times {}^H/_2 \times 64$
⬜ $C_{2,1} - C_{2,4}$	$3 \times 3 \times 64$	1	$^H/_2 \times {}^H/_2 \times 64$
🟦 $C_{3,1}$	$3 \times 3 \times 128$	2	$^H/_4 \times {}^H/_4 \times 128$
⬜ $C_{3,1} - C_{3,4}$	$3 \times 3 \times 128$	1	$^H/_4 \times {}^H/_4 \times 128$
🟦 $C_{4,1}$	$3 \times 3 \times 256$	2	$^H/_8 \times {}^H/_8 \times 256$*
🟪 $B_{4,2}$	$1 \times 1 \times 256$	1	$^H/_8 \times {}^H/_8 \times 256$
⬜ $C_{4,3} - C_{4,5}$	$3 \times 3 \times 256$	1	$^H/_8 \times {}^H/_8 \times 256$
🟧 $C_{5,1}$	$4 \times 4 \times 128$	$^1/_2$	$^H/_4 \times {}^H/_4 \times 128$*
🟪 $B_{5,2}$	$1 \times 1 \times 128$	1	$^H/_4 \times {}^H/_4 \times 128$
⬜ $C_{5,3} - C_{5,5}$	$3 \times 3 \times 128$	1	$^H/_4 \times {}^H/_4 \times 128$
🟧 $C_{6,1}$	$4 \times 4 \times 64$	$^1/_2$	$^H/_2 \times {}^H/_2 \times 64$*
🟪 $B_{6,2}$	$1 \times 1 \times 64$	1	$^H/_2 \times {}^H/_2 \times 64$
⬜ $C_{6,3} - C_{6,5}$	$3 \times 3 \times 64$	1	$^H/_2 \times {}^H/_2 \times 64$
🟧 $C_{7,1}$	$4 \times 4 \times 64$	$^1/_2$	$^H/_1 \times {}^H/_1 \times 64$
⬜ $C_{7,2}$	$4 \times 4 \times 6$	1	$^H/_1 \times {}^H/_1 \times 6$
⬜ $\mathcal{I}^{(k)}$	$3 \times 3 \times 3$	1	$^H/_1 \times {}^H/_1 \times 3$

Skip connections as temporal links. We also propose to propagate latent features between subsequent deblur blocks over time. For this, we concatenate specific layer activations from a previous iteration with some from the current deblur block. These skip connections are illustrated as green lines in Figure 3. Further, to reduce the channel dimension to match the required input shape for the next layer, we use a 1×1 convolution layer, denoted as blending layer $B_{\cdot,\cdot}$.

This way the network can learn a weighted sum by blending between the current features and propagated features from the previous iteration. This effectively halves the channel dimension. One advantage of such a construction is that we can disable these skip connections in the first deblur block and only apply these in subsequent iterations. Further, they can be applied to a pre-trained model without temporal skip connections.

Training details. Aligning inputs using homography matrices or estimated optical flow information can be error-prone and slows down the reconstruction preventing time-critical applications. Therefore, we trained the network directly on a sequence of unaligned frames featuring large camera shakes. To further challenge the network we add artificial camera shake to each blurry frame from synthetic PSF kernels on-the-fly. These PSF kernels of sizes $7 \times 7, 11 \times 11, 15 \times 15$ are generated by a Gaussian process simulating camera shake. To account for the effect of vanishing gradients, we force the output $\mathcal{I}^{(k)}$ of each deblur block to match the sharp ground-truth $\mathcal{I}^{(gt)}$ in the corresponding loss term L_k (see Figure 3).
We use ADAM [14] for minimzing the total loss $L = \sum_{k=1}^{4} L_k$ for sequences of 5 inputs. We leave the optimizer's default parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$) unchanged and use 5e-3 as the initial learning rate.

4. Experiments

We evaluate the performance of our proposed method in several experiments on challenging real-world examples. In addition, a comprehensive comparison to recent methods is given using the implementation provided by the respective authors. During inference we pass a pair of frames with resolution 720p into a deblur block iteratively. Each iteration takes approximately 0.57 seconds on an NVIDIA Titan X. For any larger frame sizes, we tile the input frames. The network was trained exclusively on our synthetically blurred dataset featuring both motion blur and camera shake. All provided results in the section are based on benchmark sets from previous methods. Our *recurrent deblur network* (RDN) generalizes to different kinds of unseen videos and recording devices. Please note, we include the full-resolution images and frames from videos in the supplementary material.

4.1. Burst Deblurring

In burst deblurring the task is to restore a sharp frame from an entire sequence of aligned images. The sequence is usually taken by a single camera and only suffers from stationary blur caused by ego-motion. In our data-driven approach, we process each observation which finally produces significantly better results than previous proposed methods, *e.g.* consider the scene provided in Figure 5. Notably, ours is the first, which is able to restore the lettering below the license plate in Figure 8. Also for the wood scene in Figure 9 sharper results are produced.

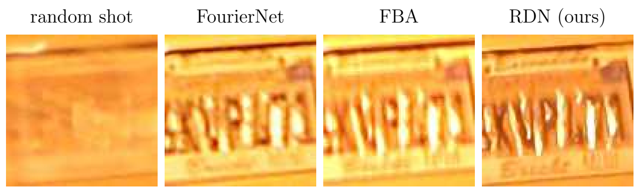

Figure 8. In contrast to previous state-of-the-art methods, our recurrent approach is able to even recover the subtle writing on the bottom of this number plate. It further reflects the original color tones from the random blurry shot.

Further, our network is applied to input images featuring spatially varying blur, which is quite common in real-world examples due to imperfect lenses or turbulences. Figure 4 shows a comparison to the Efficient Filter Flow framework (EFF) [9] which is explicitly designed to model this kind of blur. The results demonstrate two features of our approach: it is able to generalize to this kind of blur – no patch-wise processing as in [30] was necessary – and due to its recurrent nature it can deal and exploit almost arbitrary many frames for deblurring one image. However, we observe in the top row of Figure 4 that after adding more than 10 input images local contrast might saturate, potentially resulting also in a small color shift. A workaround might be some color-transfer method [30].

4.2. Video Deblurring

In contrast to the previous task, videos are usually degraded by additional blur caused by object motion. Moreover, any deblurring approach has to solve the underlying frame alignment problem. Such an alignment step can be done offline, *e.g.* using a homography matrix or by estimating optical flow fields to warp the frames to the reference frame. While this kind of preprocessing delivers an easier task to the network, it might introduce artifacts which the network later has to account for. The approach of Su *et al.* [24] (DBN) extensively use preprocessing for alignment and directly train their networks to solve both tasks: deblurring and removing artifacts. Our approach does not require any preprocessing and is therefore faster while producing comparable or better results. Figure 10 shows a comparison between the DBN in [24] and our network directly applied to the input. Significant improvement in sharpness by our method can be observed on the trousers, the hair of the woman or the hand of the baby, just to highlight a few. Artifacts due to the alignment procedure in the approach by Su *et al.* are visible on the lit wall in the Starbucks scene, for the cyclist in the second last row and in the piano scene,

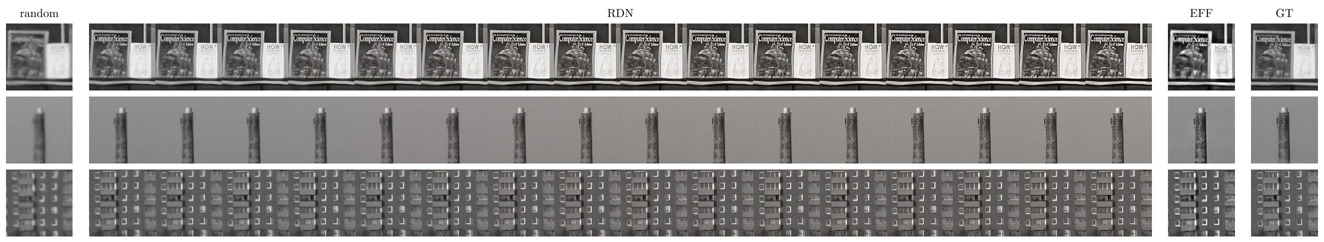

Figure 4. Recovery from image bursts with spatially varying blur. Reconstructions of a single image using 2 to 17 input frames are shown. Although the network has only seen training sequences of length 5, due to its recurrent structure it can handle longer sequences and further improve the prediction. However, too many input frames might introduce oversharpening. A random shot from the input is given on the left and the ground-truth on the right. We also compare against the EFF reconstructions from Hirsch *et al*. [9], which is dedicated to this task.

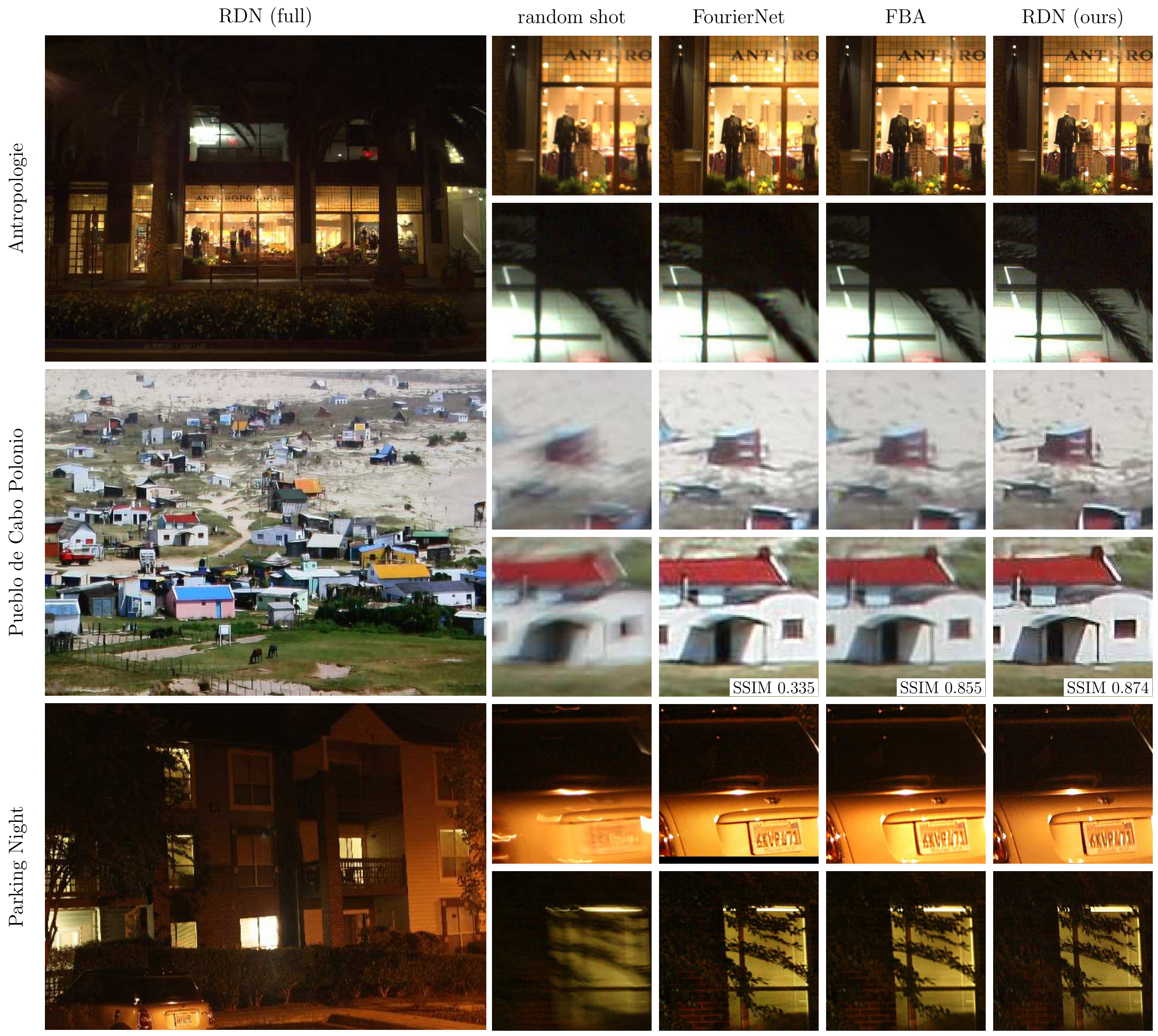

Figure 5. Comparison to state-of-the-art multi-frame blind deconvolution algorithms FourierNet [30], FBA [6] and ours (RDN) on real-world data for static scenes of low-light environments. RDN recovers significantly more detail.

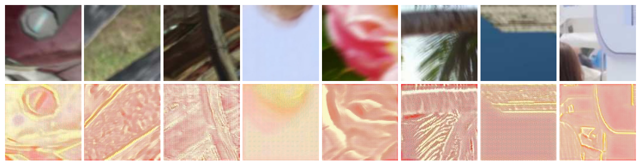
Figure 6. Visualization of features propagated along the temporal skip connections. The additional channels are projected to 2D and encoded in Hue colorspace. Apparently, the network learned to mark regions which might benefit from deblurring.

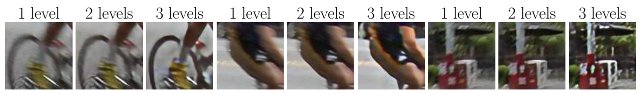

Figure 7. Multi-scale input for large motion blur. We show the deblurred results with tradition single-scale input (1 level) or extending the input sequence with upscaled version of the deblurred results at half respectively quarter resolution.

where the white keys are distorted. While ours is competitive when removing small motion, their optical flow based methods produces slightly sharper results when the camera motion is severe as seen on the road markings in the in the "bicycle" scene. Due to the limited capacity of the trained networks neither their nor our approach is fully capable of recovering the strong motion blur of very fast motion.

Using multi-scale input. While our network has been trained on sequences of constant spatial resolution only, we experimented with feeding multi-scale input to recover strong object motion. In particular, we deblurred the entire input sequence at different levels $n = 1, 2, 3$ with $1/2^{n-1}$ resolution and then up-scaled the predicted result to obtain an additional new input frame for the sequence at the higher scale. While it partly helped to deal with larger motion blur which is not covered in the training data, the upsampling can produced artifacts which the network was not trained for. Figure 7 shows such results. Although the bike became significant sharper, the static parts of the scene rendered a "comic style" appearance. Directly training such a multi-scale network seems to be an interesting research direction.

Time-structure. Our network architecture consists of an "anti-causal" structure deblurring one frame by considering the original previous frames in a sequence-to-one mapping. $\hat{\mathcal{I}} = DB(DB(\mathcal{I}, \mathcal{I}_{-1}), \ldots)$. We experimented with several sequence-to-sequence mapping approaches producing a sharp frame in an online way $\hat{\mathcal{I}}_t = DB(\mathcal{I}_t, \hat{\mathcal{I}}_{t-1})$. We noticed no learning benefit which might be caused by the limited capability of propagating temporal information.

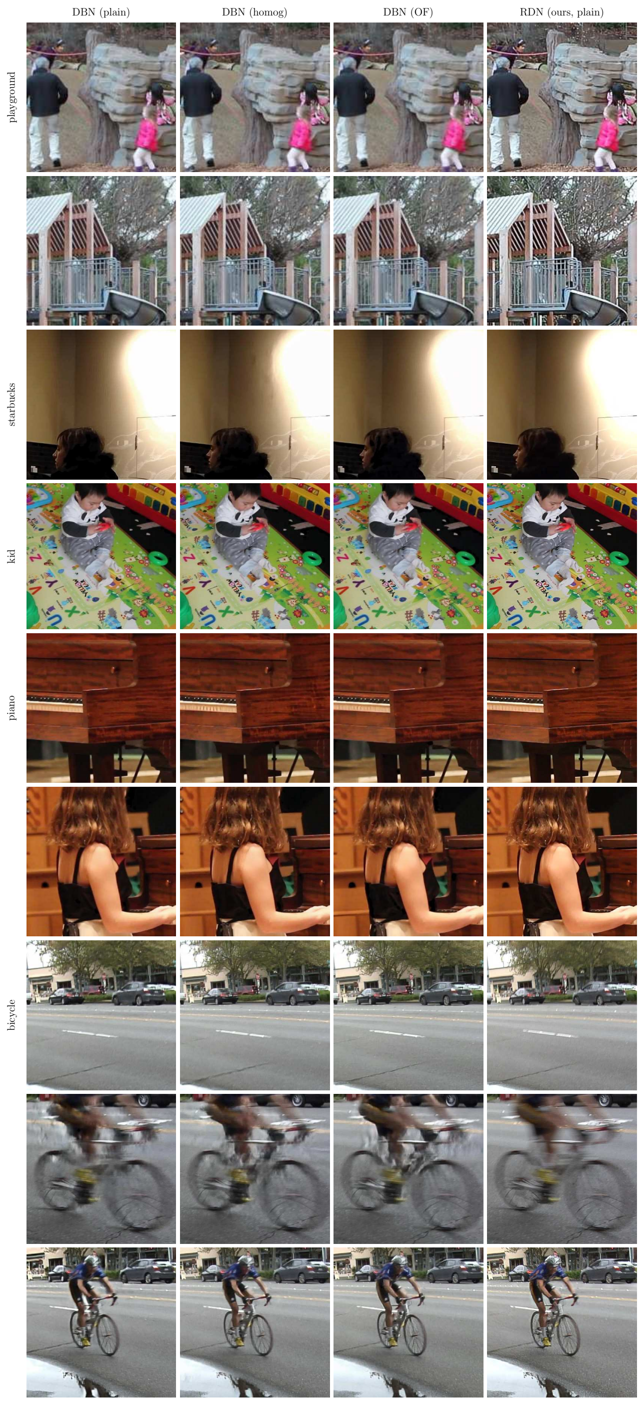

Figure 10. Video deblurring results. Applied to successive video frames with camera shake and motion blur our network produces favorable results compared to the approach by Su *et al.* [24] (DBN). While their performance depends on the type of offline preprocessing to align the individual frames (plain, homography, optical flow) our method operates directly on the unaltered input frames. As the preprocessing might introduce artifacts such as blur, smearing etc. their network learned to partially correct those, which sometimes fails. The strong motion blur of the cyclist is not correctly deblurred by any existing method.

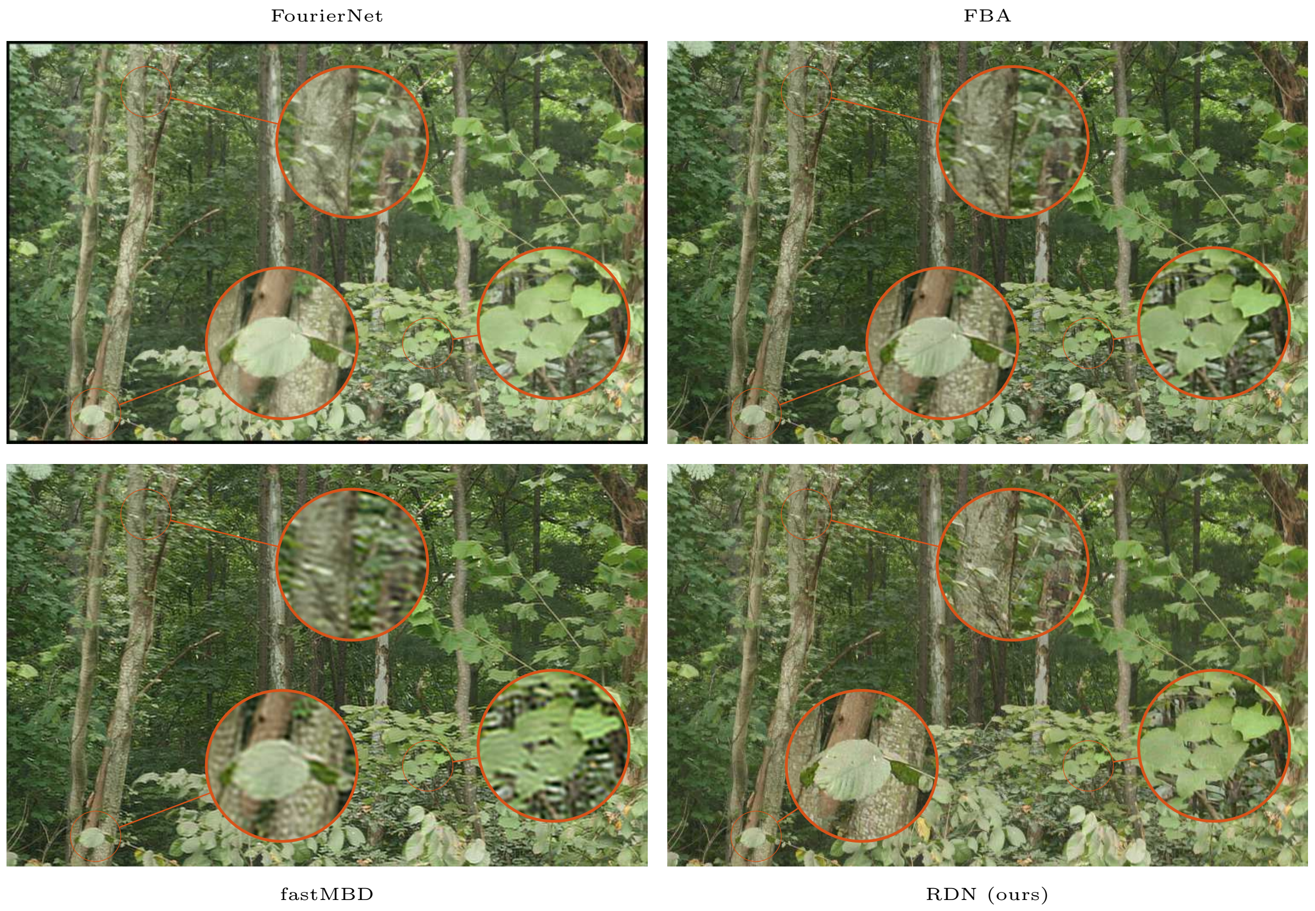

Figure 9. Performance of multi-frame blind deconvolution algorithms FourierNet [30], FBA [6] and ours (RDN) on a forest scene. The structure of the leaves and the bark of the trees is significantly sharper.

Identifying valuable temporal information. One novel feature of our designed network architecture are the temporal skip connections (Figure 3 in green) acting as information links between subsequent deblur blocks. As we do not add constraints to these links, we essentially allow the network to propagate whatever feature information seems to be beneficial for the next deblur block. To illustrate these temporal information, we visualized the respective layer activation in Figure 6. The illustration suggests that the network uses this opportunity to propagate image locations which might profit from further deblurring (yellowish parts).

5. Conclusion

We presented a novel recurrent network architecture – recurrent deblurring network (RDN) – for the efficient removal of blur caused by both ego and object motion from a sequence of unaligned blurry frames. Our proposed model enables fast processing of image sequences of arbitrary length and size. We introduce the concept of temporal skip connections between consecutive deblur blocks which allow for efficient information propagation across several time steps. Our proposed network iteratively improves the sharpness of a target frame given various blurry observations.

Furthermore, we presented a novel method for the efficient generation of a vast number of blurry/sharp video sequence pairs, which is required to train learning based methods like the one we described. Using bidirectional optical flow between consecutive frames, our method creates synthetically intermediate frames to fake high-speed video recordings. By averaging multiple consecutive frames we can emulate longer exposure times and thus motion blur in a realistic way. Hence, making use of the abundance of high-quality videos available on YouTube, we illustrated the generation process of an arbitrary amount of training data.

Acknowledgement This work was supported by the German Research Foundation (DFG): SFB 1233, Robust Vision: Inference Principles and Neural Mechanisms, TP XX.

References

[1] J.-F. Cai, H. Ji, C. Liu, and Z. Shen. Blind motion deblurring using multiple images. *Journal of computational physics*, 228(14):5057–5071, 2009. 2

[2] A. Chakrabarti. A neural approach to blind motion deblurring. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 221–235, 2016. 1, 2

[3] J. Chen, L. Yuan, C.-K. Tang, and L. Quan. Robust dual motion deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008. 2

[4] K. Cho, B. van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014. 3

[5] M. Delbracio and G. Sapiro. Burst deblurring: Removing camera shake through fourier burst accumulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2385–2393, 2015. 1

[6] M. Delbracio and G. Sapiro. Hand-held video deblurring via efficient fourier aggregation. *IEEE Transactions on Computational Imaging*, 1(4):270–283, 2015. 1, 2, 6, 8

[7] S. W. Hasinoff, K. N. Kutulakos, F. Durand, and W. T. Freeman. Time-constrained photography. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 333–340. IEEE, 2009. 2

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 4

[9] M. Hirsch, S. Sra, B. Schölkopf, and S. Harmeling. Efficient filter flow for space-variant multiframe blind deconvolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 607–614, 2010. 2, 5, 6

[10] M. Hradiš, J. Kotera, P. Zemcík, and F. Šroubek. Convolutional neural networks for direct text deblurring. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 2015–10, 2015. 2

[11] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pages 448–456, 2015. 4

[12] A. Ito, A. C. Sankaranarayanan, A. Veeraraghavan, and R. G. Baraniuk. Blurburst: Removing blur due to camera shake using multiple images. *ACM Trans. Graph.*, 3(1), 2014. 2

[13] T. H. Kim, S. Nah, and K. M. Lee. Dynamic scene deblurring using a locally adaptive linear blur model. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2016. 2

[14] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[15] C. Laurent, G. Pereyra, P. Brakel, Y. Zhang, and Y. Bengio. Batch normalized recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016. 3

[16] T. Michaeli and M. Irani. Blind deblurring using internal patch recurrence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 783–798. Springer, 2014. 2

[17] M. Noroozi, P. Chandramouli, and P. Favaro. Motion deblurring in the wild. *arXiv preprint arXiv:1701.01486*, 2017. 1, 2

[18] V. Pătrăucean, A. Handa, and R. Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *International Conference on Learning Representations (ICLR) Workshop*, 2016. 1, 3

[19] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *arXiv preprint arXiv:1505.04597*, 2015. 2, 4

[20] D. Rosenbaum and Y. Weiss. The return of the gating network: Combining generative models and discriminative training in natural image priors. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2665–2673, 2015. 2

[21] C. Schuler, H. Burger, S. Harmeling, and B. Scholköpf. A machine learning approach for non-blind image deconvolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1067–1074, 2013. 1, 2

[22] C. J. Schuler, M. Hirsch, S. Harmeling, and B. Schölkopf. Learning to deblur. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2015. 1, 2

[23] F. Šroubek and P. Milanfar. Robust multichannel blind deconvolution via fast alternating minimization. *Image Processing, IEEE Transactions on*, 21(4):1687–1700, 2012. 2

[24] S. Su, M. Delbracio, J. Wang, G. Sapiro, W. Heidrich, and O. Wang. Deep video deblurring. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 3, 5, 7

[25] J. Sun, W. Cao, Z. Xu, and J. Ponce. Learning a convolutional neural network for non-uniform motion blur removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 769–777. IEEE, 2015. 2

[26] L. Sun, S. Cho, J. Wang, and J. Hays. Edge-based blur kernel estimation using patch priors. In *IEEE International Conference in Computational Photography (ICCP)*, pages 1–8. IEEE, 2013. 2

[27] P. Svoboda, M. Hradis, L. Marsik, and P. Zemcik. Cnn for license plate motion deblurring. *arXiv preprint arXiv:1602.07873*, 2016. 2

[28] L. Wan, M. D. Zeiler, S. Zhang, Y. LeCun, and R. Fergus. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning (ICML)*, 2013. 3

[29] R. Wang and D. Tao. Recent progress in image deblurring. *arXiv preprint arXiv:1409.6838*, 2014. 2

[30] P. Wieschollek, B. Schölkopf, H. P. Lensch, and M. Hirsch. End-to-end learning for image burst deblurring. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2016. 1, 2, 3, 5, 6, 8

[31] L. Xu, J. S. Ren, C. Liu, and J. Jia. Deep convolutional neural network for image deconvolution. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1790–1798. 2014. 2

[32] H. Zhang and L. Carin. Multi-shot imaging: joint alignment, deblurring and resolution-enhancement. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2925–2932, 2014. 2

[33] H. Zhang, D. P. Wipf, and Y. Zhang. Multi-observation blind deconvolution with an adaptive sparse prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 36(8):1628–1643, 2014. 1, 2

[34] H. Zhang and J. Yang. Intra-frame deblurring by leveraging inter-frame camera motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4036–4044, 2015. 2

[35] X. Zhu, F. Šroubek, and P. Milanfar. Deconvolving psfs for a better motion deblurring using multiple images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 636–647. Springer, 2012. 2