Supplemental appendix to: Burst Image Deblurring Using Permutation Invariant Convolutional Neural Networks

Miika Aittala and Frédo Durand

Massachusetts Institute of Technology, Cambridge MA 02139, USA

1 Network architecture

The full list of layers in our architecture is shown in Table 1. Aside from the max-pooling layers and the skip connections (which refer to features from previous layers via concatenation), the architecture is essentially a linear chain of layers from the viewpoint of an individual input image track.

2 Numerical experiments

2.1 Results on Köhler et al. [1]

The dataset of Köhler et al. [1] is designed as a benchmark for deconvolution algorithms. Figure 7 in the paper shows our result and that of Wieschollek et al. [2] on their image number 2. As the dataset also contains the underlying ground truth, we evaluate the corresponding progression of SSIM value in Figure 1, for all of the four images of Köhler et al. [1]. After the sharp third frame is introduced in the input, our method reaches a high SSIM value and retains it even as low-quality input frames are later introduced.

2.2 Synthetic data

Figres 2 and 3 show results on numerical experiments with synthetic data from our training data generation pipeline (see Figure 4 in the paper), using photographs from a validation subset which was not used in training. The known ground truth allows us to quantify the quality of the solution at different burst lengths. Figure 2 shows the progression of SSIM values for 100 different bursts of 16 frames, as well as the average SSIM over them, in a manner similar to Figure 1. Figure 3 shows the expected value and standard deviation of the *change* of SSIM as each frame is introduced; ideally introducing a new frame would always result in a positive change to the SSIM.

Note that the absolute numerical values are highly dependent on the underlying dataset. The data generation pipeline is intentionally designed to generate a mixture ranging from easy to very difficult bursts, and for latter the reconstruction quality is necessarily limited. However, what can be confirmed from

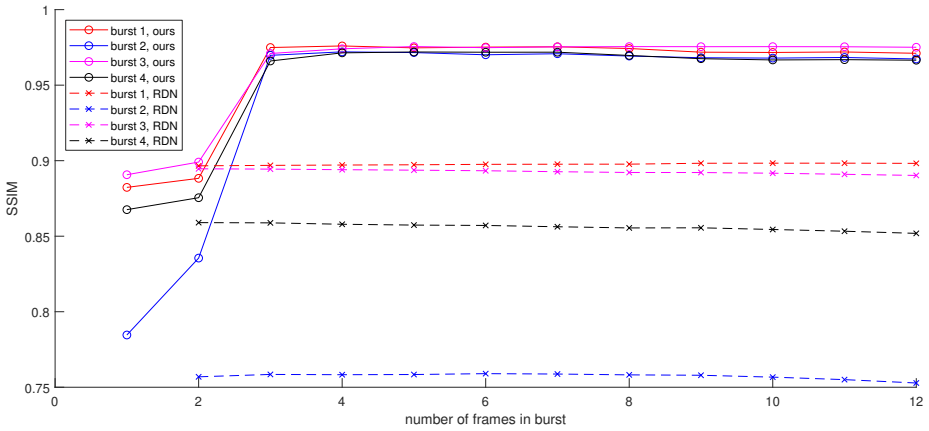

Fig. 1. Progression of SSIM value on the four images in the dataset of Köhler et al. [1] using our method and Wieschollek et al. [2].

the data is the trend that including more frames in a burst likely improves the reconstruction, and that this effect persists beyond the number of frames used during training (a per-minibatch random number of 1 to 8 images). The reconstruction tends to "saturate" in the sense that beyond some point new images are unlikely to reveal significant new information, but there is no evidence of any accumulation of artifacts with increased frame counts.

References

1. Köhler, R., Hirsch, M., Mohler, B., Schölkopf, B., Harmeling, S.: Recording and playback of camera shake: Benchmarking blind deconvolution with a real-world database. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) Computer Vision – ECCV 2012. pp. 27–40. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
2. Wieschollek, P., Hirsch, M., Schölkopf, B., Lensch, H.: Learning blind motion deblurring. In: IEEE International Conference on Computer Vision (ICCV 2017). pp. 231–240 (2017)

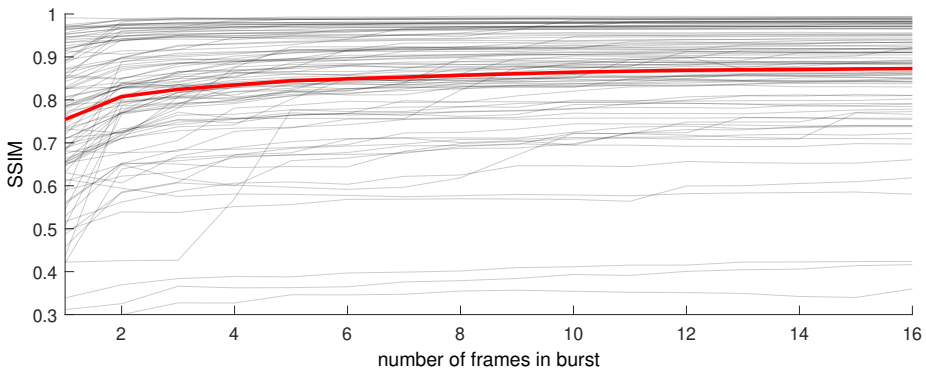

Fig. 2. Progress of SSIM value as images are added to the burst, using 100 different bursts (gray) from the synthetic validation dataset of Figure 4 in the paper. The average SSIM is plotted in red.

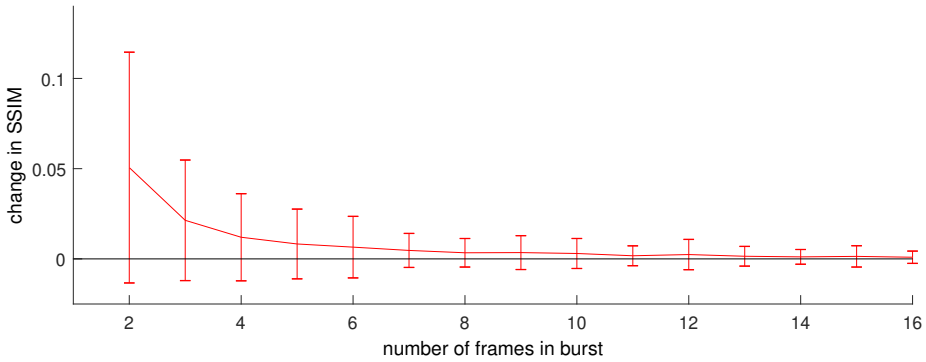

Fig. 3. The change of SSIM when a new image is added, computed over 400 bursts of synthetic data. For example, the 4th position on x-axis indicates the mean and standard deviation of the change of SSIM as a 4th image is added to a burst of 3 images. Generally, adding frames to the burst is likely to increase the quality of the reconstruction, although this effect tapers off at large burst sizes where a new image is unlikely to contain information not already present in the burst.

id	type	features	filter size	stride	activation
1	conv	64	3×3	1	ELU
2	max-pool				
3	concatenate layer 2				
4	conv	96	1×1	1	ELU
5	conv	96	4×4	2	ELU
6	max-pool				
7	concatenate layer 6				
8	conv	128	1×1	1	ELU
9	conv	128	4×4	2	ELU
10	max-pool				
11	concatenate layer 10				
12	conv	256	1×1	1	ELU
13	conv	256	4×4	2	ELU
14	max-pool				
15	concatenate layer 14				
16	conv	384	1×1	1	ELU
17	conv	384	4×4	2	ELU
18	conv	384	3×3	1	ELU
19	deconv	384	4×4	2	ELU
20	max-pool				
21	concatenate layers 17, 20				
22	conv	384	1×1	1	ELU
23	conv	384	3×3	1	ELU
24	deconv	256	4×4	2	ELU
25	max-pool				
26	concatenate layers 13, 25				
27	conv	256	1×1	1	ELU
28	conv	256	3×3	1	ELU
29	deconv	192	4×4	2	ELU
30	max-pool				
31	concatenate layers 9, 30				
32	conv	192	1×1	1	ELU
33	conv	192	3×3	1	ELU
34	deconv	96	4×4	2	ELU
35	max-pool				
36	concatenate layers 1, 35				
37	conv	96	1×1	1	ELU
38	conv	96	3×3	1	ELU
39	max-pool				
40	conv	64	3×3	1	ELU
41	conv	3	3×3	1	linear

Table 1. The sequence of layers in our network architecture. max-pool indicates a max-pooling across different input tracks as described in the main document (*not* the spatial reduction often referred to by the same name). The layers prior to the dividing line are evaluated separately for each input track, after which the final max-pooling combined with the last two layers compute a single output for the entire set.