Supplemental Material for FlexISP: A Flexible Camera Image Processing Framework

Felix Heide[1,2] Markus Steinberger[1,3] Yun-Ta Tsai[1] Mushfiqur Rouf[1,2] Dawid Pająk[1] Dikpal Reddy[1]
Orazio Gallo[1] Jing Liu[1,4] Wolfgang Heidrich[2,5] Karen Egiazarian[1,6] Jan Kautz[1] Kari Pulli[1]
[1]NVIDIA [2]UBC [3]TU Graz [4]UC Santa Cruz [5] KAUST [6] TUT

1 Flexibility, Code Reuse, and Implementation Details

In Algorithm 1 we show the pseudo-code of our system. This pseudo-code can be readily converted to Matlab code. **Note that switching the application essentially corresponds to replacing the matrix A that is given to the system, yielding a very flexible system**, details are below.

Algorithm 1: Algorithm of our unified pipeline.

Data: z, $\mathbf{A}$, $\mathbf{M}$, ϕ_0, ϕ_1, ϕ_2, $\gamma = 40$, $\theta = 1$, N
Result: x
$\mathbf{K}_0 = \nabla$, $\mathbf{K}_1 = \mathbb{I}$, $\mathbf{K}_2 = \begin{bmatrix} \mathbf{K}_{2R} & 0 & 0 \\ 0 & \mathbf{K}_{2G} & 0 \\ 0 & 0 & \mathbf{K}_{2B} \end{bmatrix}$, $\mathbf{K} = [\mathbf{K}_0^T \mathbf{K}_1^T \mathbf{K}_2^T]$;
$\bar{x}^0 = x^0 = \text{initial\_guess}(z)$;
$y^0 = \mathbf{K}x^0$;
$\bar{z} = \mathbf{M}z$;
$\tau = 0.9/(\gamma \parallel \mathbf{K} \parallel^2)$; `// Ensure` $\gamma\tau \parallel \mathbf{K} \parallel^2 < 1$
$k = 0$;
while $k < N$ **do**
  $y^{k+1} = \text{prox}_{\gamma F^*}(y^k + \gamma \mathbf{K}\bar{x}^k; \phi_0, \phi_1, \phi_2)$;
  $x^{k+1} = \text{prox}_{\tau G}(x^k - \tau \mathbf{K}^T y^{k+1}; \bar{z}, \mathbf{A})$;
  $\bar{x}^{k+1} = x^{k+1} + \theta(x^{k+1} - x^k)$;
  $k = k + 1$;

In the algorithm 1 above the cross-channel matrix K_1 contains $\mathbf{K}_{2R}$, which, when multiplied with the red channel, computes the cross-channel gradients for it:

$$\mathbf{K}_{2R} = \begin{bmatrix} \beta_{RG}\left(\text{diag}(\mathbf{x}_G)\nabla_x - \text{diag}(\nabla_x \mathbf{x}_G)\right) \\ \beta_{RG}\left(\text{diag}(\mathbf{x}_G)\nabla_y - \text{diag}(\nabla_y \mathbf{x}_G)\right) \\ \beta_{RB}\left(\text{diag}(\mathbf{x}_B)\nabla_x - \text{diag}(\nabla_x \mathbf{x}_B)\right) \\ \beta_{RB}\left(\text{diag}(\mathbf{x}_B)\nabla_y - \text{diag}(\nabla_y \mathbf{x}_B)\right) \end{bmatrix}_{4n \times n} \quad \text{and } \mathbf{K}_{2G}, \mathbf{K}_{2B} \text{ analogously.} \tag{1}$$

Data term weight There are different ways to weight regularization versus the data term. When solving for $\text{prox}_{\tau G}$, see Eq. 6 and 7 in the paper, we scale the matrix $\mathbf{A}$ by the (application-specific) data-term weight $\frac{1}{\eta}$:

$$\left(\frac{\tau}{\eta}\mathbf{A}^T\mathbf{A} + \mathbb{I}\right)\mathbf{x} = \left(\frac{\tau}{\eta}\mathbf{A}^T\mathbf{z} + \mathbf{v}\right). \tag{2}$$

CG and on-demand computation of A CG does not require the full matrix $\mathbf{A}$ to be in memory, but only requires particular entries to be evaluated. So in our implementation, rather than explicitly expressing the full matrix $\mathbf{A}$, we evaluate it procedurally where needed. This allows us to exploit its structure, i.e., as a series of convolutions (using a convolution kernel, instead of explicitly building the convolution matrix $\mathbf{B}$) and per-pixel operators (decimation $\mathbf{D}$, mask matrix $\mathbf{M}$, or warp $\mathbf{S}$, expressed as procedural functions). Therefore, the representation itself is very compact, and there is no need to store the full matrix in memory.

Calibration of A Depending on the application, the matrix $\mathbf{A}$ maybe slightly inaccurate; e.g., the blur matrix might be imperfect even when calibrated from data [Xu and Jia 2010]. Fortunately, given our powerful priors, FlexISP is not very sensitive to such slight inaccuracies. For instance, for the color array camera we simply assume a 2×2 box blur, yet no artifacts are introduced.

1.1 "Plug-and-play" Applications

Switching to a different application simply means plugging in a different matrix A and M (data term) from Table 1 into Algorithm 1 and changing a small set of parameters, see below, due to application-specific noise, chromatic aberrations, and so forth. In our optimization scheme, where the data-term proximal operator is computed via conjugate gradients, the observation matrix can be formulated in a *matrix-free,*

procedural way, rather than be explicitly created. Therefore, one simply needs to provide functions that compute the matrix and its transpose. The "plug-and-play" manner of our system is therefore also reflected in the code itself.

Application	A	M
Demosaicking	D	$\mathbb{I}$
Deblurring	B	$\mathbb{I}$
iHDR	MDB	reliable pixels
Color Array	$M_i DBS_i$	reliable flow
Burst	$M_i DS_i$	good alignment
JPEG Deblocking	$MDBJ^{-1}$	$\mathbb{I}$

Table 1: *Configuring the matrices for our different applications.*

The data term has different characteristics in different applications (noise characteristics, etc.), but a small set of parameters (stated at the very top of Algorithm 1 is enough to handle these data-specific changes—no image-dependent modifications are necessary.

Parameters Table 2 shows the parameters which we have optimized on a representative training set of images. The change of matrix A and these few parameters then completely define our algorithm for each application.

Application	ϕ_0 (TV)	ϕ_1 (BM3D)	ϕ_2 (cross)	η (inverse data)
Demosaicking	0.1	1.0	0.25	0.002
Deblur	0.2	1.0	1.0	0.001
iHDR	0.1	1.0	0.1	0.001
Color Array	0.1	1.0	0.25	0.01
Burst	0.15	1.0	0.1	2.0
JPEG Deblocking	0.05	1.0	0.0	0.0005

Table 2: *Optimization weights for our applications.*

2 Convergence of Optimization

We present convergence plots for two different examples in Figure 1. The first example is demosaicking of a Bayer image, where we initialize our optimization with: $\mathbf{x}^0 = 0$, $\mathbf{x}^0 =$ average of neighboring pixels, $\mathbf{x}^0 =$ Malvar [2004] demosaicking. The second example is reconstructing an image from the interlaced HDR sensor. We use three different initializations: $\mathbf{x}^0 = 0$, $\mathbf{x}^0 =$ long exposure, $\mathbf{x}^0 =$ short exposure, $\mathbf{x}^0 =$ mean of long and short exposure. Notice how in all cases, we converge to the same low final error (or conversely the same high PSNR), no matter how we initialized the optimization. This indicates, that in practice our optimization converges, even though there are no theoretical guarantees due to our use of non-convex priors.

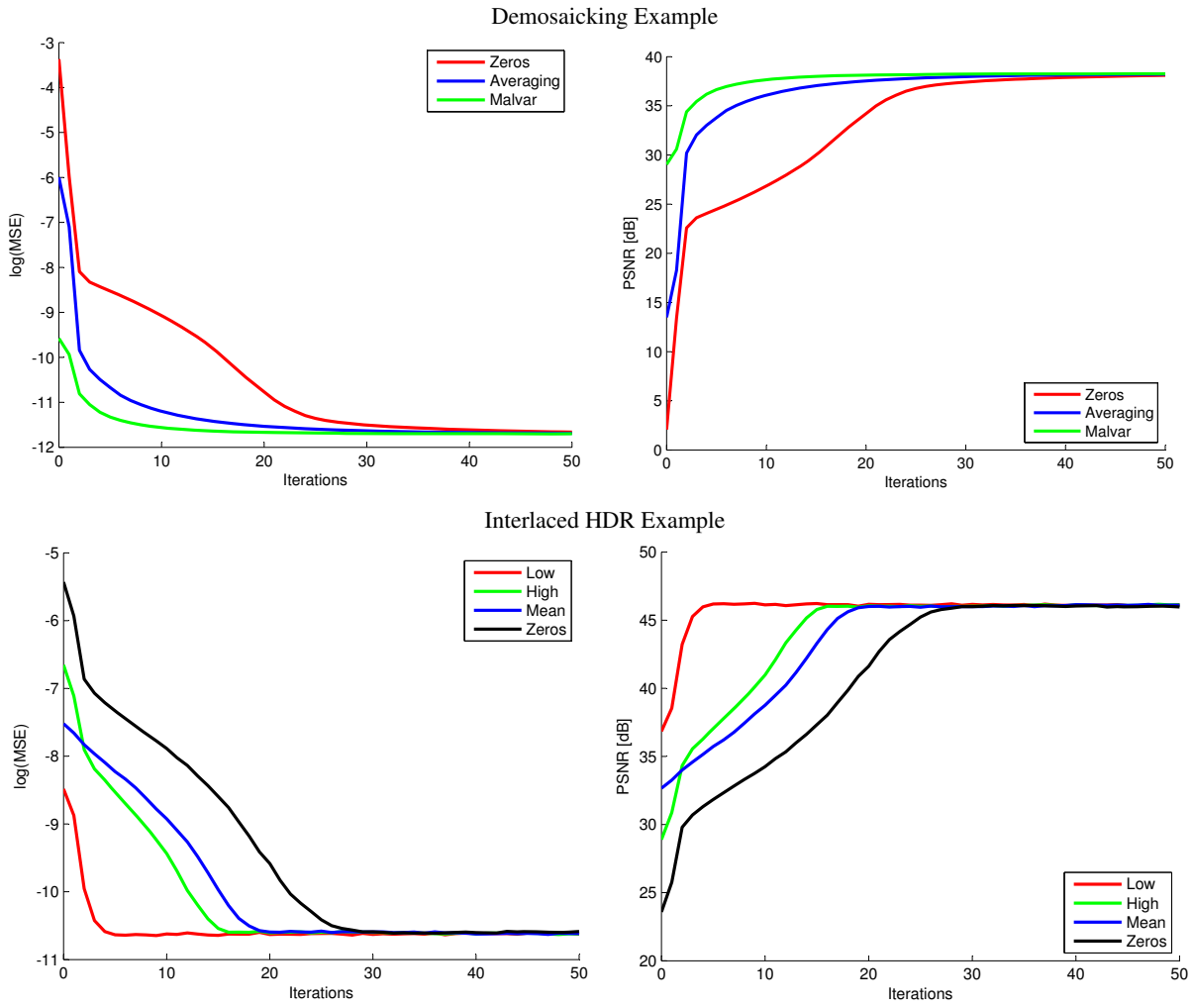


Figure 1: *Effects of different initial iterates $\boldsymbol{x}^0$ on the convergence for a simple synthetic demosaicking example **(top)** and an interlaced HDR example with ground truth captured through a bracketed sequence **(bottom)**. We compare here the convergence plots for different initial iterates (which are described in the text in detail). These plots are given here for (log of) mean squared error **(left)** and PSNR **(right)**. While a naive initialization requires 30 or so iterations, with a good initialization 4-5 iterations suffice for good results.*

There are examples where our optimization might not converge (but neither would other methods). For instance, if the set of unknowns is large enough, even with the priors there are many possible solutions. This can occur, e.g., if the image data is missing from so large an area that the self-similarity prior cannot recover the signal, and the result of any method strongly depends on the initial guess. In those cases even global gradient priors would not help much, and still stronger priors would be needed.

2.1 Accelerating Convergence Using TV

We motivated the choice of TV as an external prior by it making the problem more convex. This convexification is a core benefit of using TV in addition to BM3D, and can give gains over just using BM3D as a prior. In Figure 2 we quantify this effect with convergence plots for the interlaced HDR example from Table 3 of the main paper.

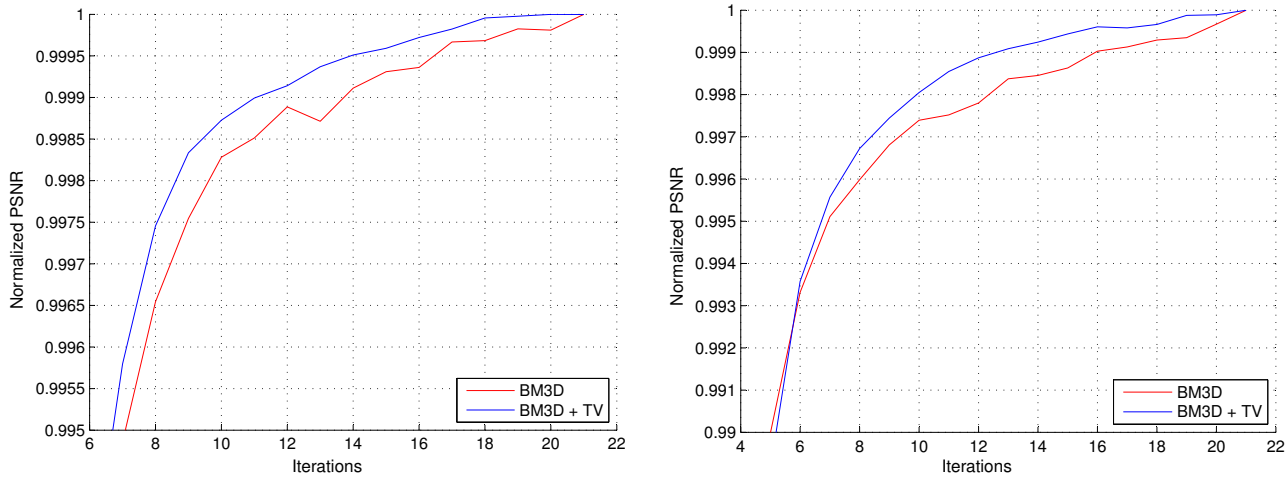


Figure 2: *Convergence for the comparison in Table 3 (main paper) with $\sigma = 1.83$. The x-axis represents iterations and the y-axis represents the quality of the estimate at a given iteration.*

The plots in Figure 2 demonstrate that adding TV stabilizes the solver so that it converges faster than just using the non-linear non-convex BM3D prior.

3 Demosaicking

3.1 Pure Demosaicking

In Table 3 we show the PSNR values for all images (and for each channel) of the McMaster color image dataset [Zhang et al. 2011]. We compare our results against recent state-of-the-art demosaicking techniques: SOLC, AHD, SA, DLMMSE, SSD, LDI-NLM and LDI-NAT. See Zhang et al. [2011] for a description of each method. Note that several demosaicking example images are included in the supplemental web page. We show visual comparisons on selected image patches of both the McMaster dataset and real-world examples in Figure 3.

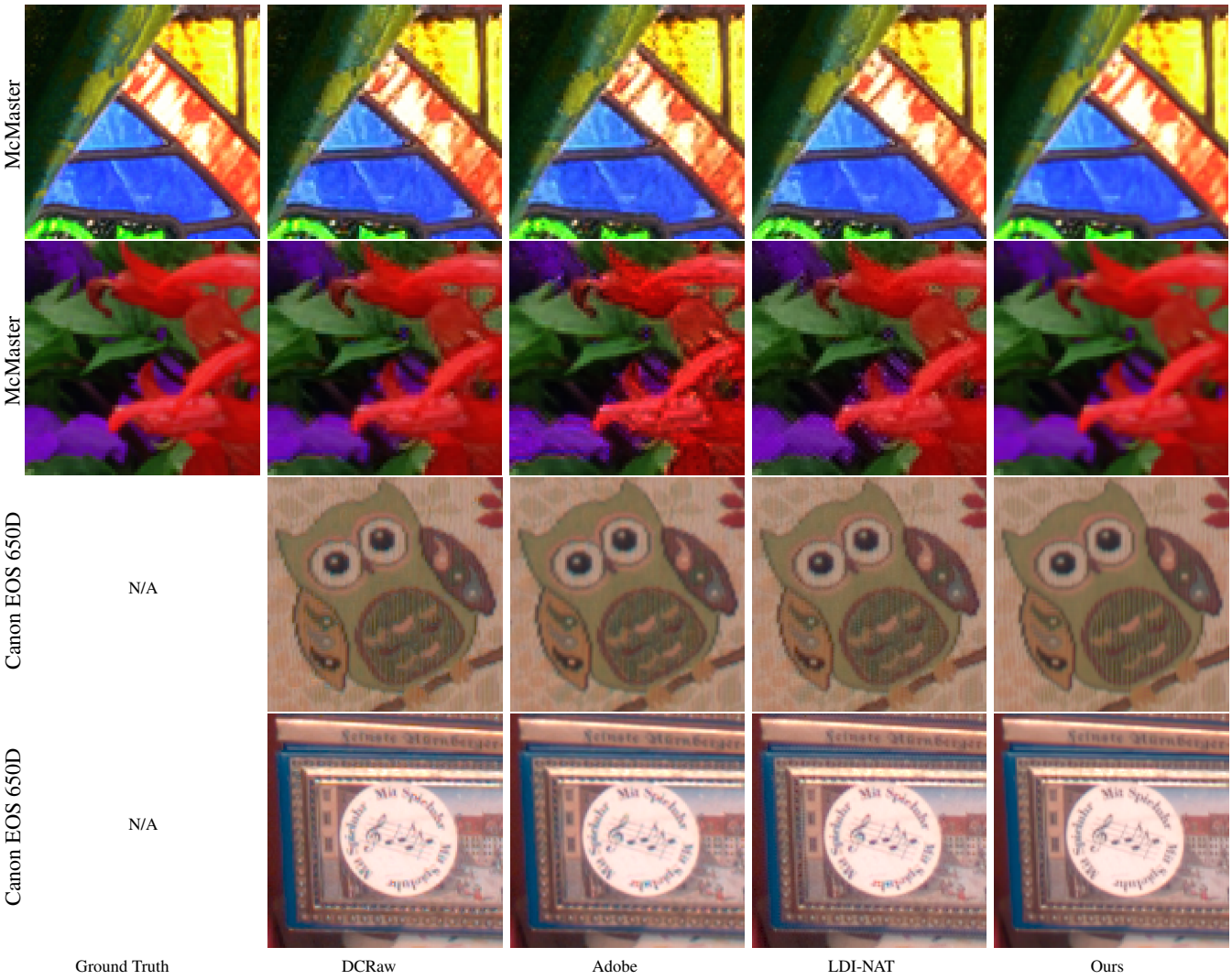


Figure 3: *Demosaicking examples. Note how our method produces results virtually indistinguishable from ground truth (top two rows), whereas competing methods contain visible artifacts (please zoom in to the pdf). In the two real-world examples at the bottom one can see artifacts in the images computed with competing methods (structural patterns, color fringing). Our method produces clean images that are more truthful to the original scene (via visual inspection of the actual objects, since ground truth is not available for these real-world captures).*

Image	Channel	SOLC	AHD	SA	DLMMSE	SSD	LDI-NLM	LDI-NAT	Photoshop CameraRAW	DCRAW	Ours
1	R	28.26	26.02	23.53	26.94	27.28	28.81	29.29	27.47	27.81	**31.07**
	G	31.22	29.82	25.17	30.63	30.68	32.31	32.67	30.39	31.68	**33.93**
	B	26.34	24.04	22.05	24.82	25.12	26.47	26.71	26.13	25.55	**28.85**
2	R	33.68	32.47	31.63	33.30	33.61	34.66	35.02	32.44	33.92	**35.78**
	G	37.62	37.20	34.00	37.66	37.81	39.01	39.08	37.89	38.55	**40.52**
	B	32.11	31.26	30.74	31.86	32.01	32.79	32.92	32.20	32.13	**33.79**
3	R	30.64	31.10	31.47	32.60	32.81	33.41	33.05	31.25	32.00	**36.52**
	G	33.73	33.49	32.75	35.28	35.05	35.50	35.51	33.92	34.22	**38.84**
	B	28.60	29.67	29.80	30.70	30.93	30.99	30.31	27.45	29.83	**33.16**
4	R	32.80	33.76	34.59	34.70	36.36	37.41	36.25	33.20	36.20	**41.63**
	G	37.16	35.66	34.05	36.99	38.98	39.01	40.33	39.53	39.79	**44.49**
	B	30.89	31.48	32.19	32.07	33.49	34.02	33.30	30.40	33.35	**37.71**
5	R	33.61	29.52	28.60	30.38	31.10	34.50	35.05	28.53	32.32	**38.80**
	G	36.28	34.73	30.97	35.11	35.43	37.67	38.15	36.09	37.05	**40.87**
	B	30.47	28.78	28.08	29.41	29.48	31.02	31.16	30.30	30.71	**32.91**
6	R	37.14	33.92	32.23	34.98	36.09	38.59	39.40	36.38	37.01	**41.47**
	G	40.30	37.72	32.50	38.61	38.85	41.70	43.42	41.44	41.25	**45.64**
	B	34.00	29.96	29.14	31.15	31.72	34.21	34.97	34.28	33.08	**38.04**
7	R	33.85	35.64	37.03	38.30	36.61	36.28	36.09	33.58	34.44	**40.86**
	G	36.34	37.36	40.39	40.70	37.62	37.66	37.41	36.19	36.06	**42.80**
	B	32.45	35.07	36.22	37.29	36.38	34.59	34.49	33.52	33.42	**38.69**
8	R	34.87	34.15	35.31	35.45	35.31	36.89	36.31	33.60	36.23	**39.98**
	G	39.09	39.45	38.49	41.43	40.34	40.44	40.29	38.45	39.46	**43.97**
	B	35.04	35.79	35.82	36.99	36.76	36.84	36.67	35.72	36.04	**39.42**
9	R	34.36	31.54	30.71	32.39	33.72	35.54	35.49	30.88	34.72	**38.62**
	G	39.62	37.99	33.83	38.73	39.52	41.56	41.73	40.46	41.14	**43.44**
	B	35.34	34.00	32.54	34.66	35.38	36.54	36.30	34.54	35.74	**38.15**
10	R	36.86	33.99	34.03	34.70	36.33	37.64	38.26	27.90	36.00	**40.36**
	G	40.86	39.17	36.15	40.00	40.23	42.19	42.64	40.08	41.41	**44.18**
	B	36.08	34.88	34.78	35.55	36.13	36.51	36.83	35.10	36.05	**38.66**
11	R	38.12	36.13	36.16	36.91	38.16	39.25	39.82	31.65	37.72	**41.28**
	G	40.78	39.34	37.11	40.44	40.19	41.66	42.57	39.56	41.01	**43.52**
	B	37.19	34.73	34.33	35.75	36.81	37.50	37.66	37.22	36.19	**39.84**
12	R	37.13	33.60	34.49	34.74	35.37	37.62	38.36	27.08	35.65	**41.54**
	G	40.17	40.09	37.66	39.59	39.70	41.45	41.49	39.19	40.70	**44.11**
	B	35.70	36.24	36.24	36.47	37.11	37.51	37.59	35.47	36.81	**39.01**
13	R	39.80	37.91	38.11	38.66	40.01	42.23	41.77	34.92	40.02	**44.02**
	G	43.46	42.16	39.90	42.57	43.82	45.55	44.89	42.58	44.53	**45.83**
	B	37.65	36.20	36.51	36.75	37.19	37.88	38.13	37.24	37.55	**39.09**
14	R	37.85	37.33	36.82	37.74	38.66	39.28	39.39	34.43	38.67	**40.71**
	G	41.37	40.65	38.79	41.13	41.93	42.62	42.84	41.16	42.24	**44.22**
	B	35.64	34.30	34.45	34.78	35.00	35.82	36.12	35.34	35.48	**37.01**
15	R	36.44	34.88	34.87	35.32	36.23	37.34	36.95	29.50	36.29	**38.35**
	G	41.20	40.27	38.13	40.71	40.75	42.39	42.68	41.40	41.90	**43.27**
	B	38.17	36.84	36.52	37.30	37.90	38.49	38.99	38.54	38.00	**39.67**
16	R	32.75	30.95	28.75	31.95	32.21	34.18	34.97	32.06	32.59	**34.90**
	G	34.09	32.36	28.60	33.22	32.99	35.00	35.59	32.62	34.13	**36.19**
	B	31.63	26.85	24.87	28.06	28.30	31.12	31.53	**34.44**	29.43	34.31
17	R	31.24	27.12	25.35	28.32	29.24	31.60	32.14	25.56	30.53	**33.88**
	G	35.17	32.13	26.68	33.31	33.62	37.31	37.62	36.79	36.55	**39.77**
	B	30.69	26.65	25.06	27.77	28.38	30.78	30.91	28.69	29.57	**33.39**
18	R	32.69	32.30	31.61	33.32	33.24	34.63	34.58	32.52	33.23	**35.10**
	G	36.20	35.69	33.84	37.02	35.91	37.30	37.27	36.30	35.95	**37.41**
	B	33.43	31.90	31.11	32.93	33.44	34.87	34.30	33.44	33.24	**36.78**
AVG	R	34.71	33.05	32.68	34.06	34.71	36.10	36.23	31.27	34.74	**38.60**
	G	38.11	37.10	34.63	38.10	38.08	39.46	39.79	38.00	38.76	**41.83**
	B	33.41	32.30	31.87	33.15	33.47	34.33	34.38	33.33	33.46	**36.58**
AVG	RGB	35.41	34.15	33.06	35.10	35.42	36.63	36.80	34.20	35.65	**39.01**

Table 3: *PSNR values for demosaicking the McMaster dataset. The bold numbers indicate the best method.*

3.2 Joint Demosaicking and Denoising

In Table 4 we compare our method for reconstructing Bayer images that contain noise to other state-of-the-art methods, as described in [Jeon and Dubois 2013]. Note, that we did not change our method from Section 3 to account for noise, as it naturally handles this case. Our method beats all state-of-the-art methods in virtually every case. Figure 4 shows a few examples for comparison.

No.	σ	JDD	Zhang1	LASIP	Zhang2	Condat 9x9 BM3D	Condat 13x13 BM3D	GT 5x5 BM3D	GT 7x7 BM3D	GT 9x9 BM3D	GT 11x11 BM3D	GT 13x13 BM3D	GT 15x15 BM3D	GT 11x11 h_{LN}	Ours
1	0	38.55	39.43	33.44	36.35	36.91	37.65	36.56	37.99	38.50	38.81	38.95	39.01	38.81	**40.68**
		1.154	1.047	1.907	1.271	1.427	1.362	1.280	1.175	1.132	1.113	1.104	1.100	1.113	**0.786**
	4	34.73	35.14	31.89	34.02	34.34	34.68	33.89	34.83	35.09	35.21	35.29	**35.33**	33.81	35.19
		1.768	1.630	2.238	1.711	1.912	1.858	1.867	1.727	1.689	1.672	1.662	1.656	1.763	**1.329**
	8	31.14	31.41	29.78	30.97	31.37	31.54	30.73	31.46	31.65	31.72	31.77	31.79	30.56	**31.93**
		2.667	2.475	2.798	2.413	2.619	2.562	2.802	2.546	2.469	2.440	2.423	2.413	2.627	**1.927**
	12	28.84	29.09	28.17	28.72	29.31	29.46	28.53	29.23	29.44	29.52	29.57	29.59	28.45	**30.09**
		3.565	3.310	3.335	3.144	3.322	3.220	3.782	3.369	3.225	3.159	3.125	3.107	3.446	**2.476**
	16	27.21	27.49	26.94	27.10	27.84	28.04	26.98	27.69	27.92	28.04	28.09	28.12	26.99	**28.74**
		4.442	4.104	3.856	3.834	4.002	3.825	4.759	4.165	3.943	3.821	3.761	3.725	4.198	**3.004**
	20	25.95	26.29	25.97	25.87	26.76	27.01	25.83	26.55	26.81	26.97	27.04	27.08	25.92	**27.65**
		5.300	4.862	4.360	4.472	4.654	4.385	5.723	4.934	4.623	4.439	4.344	4.281	4.895	**3.515**
8	0	35.98	36.62	32.19	34.43	33.07	33.47	34.04	35.30	35.68	35.85	35.92	35.95	35.85	**38.63**
		1.395	1.315	2.165	1.542	1.933	1.893	1.571	1.479	1.435	1.424	1.419	1.417	1.424	**0.967**
	4	33.51	33.85	31.11	32.84	32.02	32.32	32.47	33.45	33.72	33.82	33.88	33.92	31.53	**35.19**
		1.988	1.875	2.510	1.951	2.360	2.319	2.114	2.008	1.966	1.951	1.944	1.941	2.115	**1.449**
	8	30.67	30.82	29.37	30.45	30.51	30.75	30.36	31.16	31.38	31.46	31.51	31.54	28.84	**32.48**
		2.873	2.724	3.099	2.643	2.969	2.914	2.934	2.756	2.691	2.669	2.654	2.646	3.014	**2.015**
	12	28.57	28.68	27.86	28.52	29.16	29.40	28.63	29.40	29.62	29.71	29.77	29.80	26.77	**30.61**
		3.768	3.569	3.700	3.362	3.579	3.484	3.769	3.495	3.378	3.334	3.306	3.289	3.919	**2.569**
	16	26.97	27.11	26.61	27.00	28.00	28.27	27.26	28.02	28.28	28.39	28.46	28.49	25.24	**29.15**
		4.656	4.371	4.295	4.055	4.187	4.023	4.577	4.209	4.031	3.952	3.900	3.873	4.784	**3.116**
	20	25.68	25.88	25.57	25.76	27.01	27.33	26.16	26.92	27.20	27.35	27.42	27.46	24.07	**27.94**
		5.533	5.146	4.872	4.719	4.794	4.546	5.344	4.893	4.657	4.536	4.457	4.412	5.609	**3.660**
19	0	40.05	41.13	37.15	38.99	37.99	38.71	39.37	40.42	40.78	40.93	41.00	41.02	40.93	**41.76**
		0.815	0.763	1.212	0.904	0.969	0.932	0.831	0.777	0.755	0.749	0.746	0.746	0.749	**0.723**
	4	36.08	36.56	34.70	36.00	35.52	35.87	35.84	36.51	36.68	36.73	36.77	36.78	34.84	**37.18**
		1.490	1.383	1.714	1.427	1.564	1.539	1.533	1.435	1.413	1.408	1.406	1.404	1.522	**1.241**
	8	32.73	33.21	32.22	32.88	33.09	33.29	32.62	33.35	33.55	33.61	33.65	33.66	31.44	**34.19**
		2.363	2.130	2.300	2.084	2.251	2.209	2.451	2.233	2.153	2.126	2.117	2.113	2.404	**1.779**
	12	30.54	31.10	30.58	30.76	31.39	31.63	30.49	31.34	31.63	31.76	31.80	31.83	29.38	**32.34**
		3.210	2.834	2.777	2.679	2.889	2.778	3.338	2.979	2.821	2.745	2.714	2.697	3.184	**2.276**
	16	28.92	29.60	29.35	29.25	30.16	30.50	29.01	29.96	30.32	30.52	30.61	30.65	27.97	**30.90**
		4.033	3.500	3.197	3.209	3.488	3.281	4.172	3.676	3.439	3.304	3.235	3.196	3.886	**2.762**
	20	27.61	28.44	28.35	28.04	29.20	29.65	27.90	28.89	29.33	29.60	29.72	**29.79**	26.92	29.68
		4.840	4.138	3.593	3.697	4.065	3.753	4.963	4.334	4.015	3.826	3.717	3.647	4.532	**3.251**
24	0	35.48	35.49	33.70	34.51	35.21	35.29	35.01	35.57	35.62	35.68	35.71	35.72	35.68	**36.99**
		1.022	1.122	1.601	1.168	1.071	1.055	1.043	1.001	0.973	0.969	0.968	0.967	0.969	**0.887**
	4	33.65	33.58	32.40	33.12	33.72	33.77	33.34	33.88	33.93	33.96	33.98	34.00	32.88	**34.90**
		1.680	1.755	2.125	1.668	1.686	1.679	1.727	1.651	1.634	1.631	1.627	1.626	1.727	**1.473**
	8	31.28	31.24	30.53	31.01	31.70	31.76	31.07	31.66	31.77	31.81	31.83	31.84	29.94	**32.37**
		2.547	2.554	2.794	2.359	2.421	2.401	2.633	2.450	2.396	2.379	2.369	2.365	2.630	**2.110**
	12	29.37	29.46	29.03	29.22	30.03	30.14	29.23	29.89	30.05	30.13	30.16	30.18	27.94	**30.66**
		3.425	3.315	3.379	3.050	3.120	3.045	3.536	3.223	3.104	3.051	3.025	3.012	3.448	**2.686**
	16	27.87	28.08	27.87	27.82	28.71	28.89	27.81	28.51	28.73	28.85	28.90	28.93	26.56	**29.32**
		4.294	4.050	3.901	3.694	3.795	3.636	4.407	3.965	3.773	3.673	3.618	3.590	4.186	**3.233**
	20	26.64	26.97	26.95	26.68	27.64	27.90	26.69	27.41	27.66	27.84	27.90	27.95	25.56	**28.17**
		5.154	4.757	4.372	4.292	4.450	4.192	5.240	4.679	4.412	4.259	4.170	4.120	4.861	**3.771**

Table 4: *PSNR and S-CIELAB values for the Kodak dataset containing Gaussian noise with varying σ. The bold numbers indicate the best method. See [Jeon and Dubois 2013] for a description of the competing methods.*

(a) Ground Truth **(b)** [Jeon and Dubois 2013] (GT 15×15) **(c)** Ours

Figure 4: *Joint demosaicking and denoising comparison (Kodak dataset). Please zoom into the PDF. In terms of PSNR, we achieve better results in all cases. With our method (c), the house image is visually crisper and has no low-frequency color noise, unlike in (b). In fact, our result is almost too crisp, lowering the weight for the data term would yield a smoother result (see Fig. 14 for a related discussion).*

4 Deconvolution

We compare our method against several other state-of-the-art methods, following the procedure presented by Schuler et al. [2013]. The images and blur kernels used in this comparison are visualized in Figure 5. The PSNR for each image and each technique are given in Table 5. The average PSNR can be found in the main paper. The images can also be inspected in the supplemental web-pages. Figure 6 shows a few results comparing our technique with other state-of-the-art methods. Even more examples can be found in the supplemental web-pages.

Figure 5: *The dataset used for comparing our deconvolution to 4 other state-of-the-art methods. The procedure follows Schuler et al. [2013]. We achieve better deblurring quality than these previous methods. The full PSNR table can be found in Table 5.* The deconvolution results can be browsed in the supplemental web-pages.

Figure 6: *We first compare deconvolution of two artificially blurred color images (using measured real-world camera shake [Levin et al. 2007]). As can be seen, our method produces the least artifacts (e.g., see smooth backgrounds). In the last row, we show a real-world example from Levin et al.'s [2007] database. Similarly, our method produces the least artifacts. Please zoom into the PDF. (Images courtesy of Wikipedia user Hans Hillewaert and Pierre Dalous, and [Levin et al. 2009].)*

	Method	1	2	3	4	5	6	7	8	9	10	11
Case (a)	Blurred	20.8004	22.9252	22.6989	20.9770	22.2170	22.7570	21.9489	17.0763	22.3771	22.1460	23.2252
	Levin et al. [2007]	23.2871	26.6641	26.2106	22.7202	24.8653	25.7623	24.2881	17.4262	24.8001	24.1561	25.7881
	Krishnan et al. [2009]	23.4305	26.6857	26.3295	22.8015	24.9406	25.7638	24.3204	17.4299	24.8236	24.1149	25.7402
	IDD-BM3D [2012]	24.0598	**28.2769**	26.9892	23.0866	25.6261	26.3341	24.4639	**17.4885**	25.1974	24.3742	26.0144
	MLP [2013]	24.2612	27.9809	26.9226	23.2296	25.7352	26.4476	24.5986	17.4818	25.2033	24.4735	26.1373
	Ours	**24.3327**	28.1724	**27.0898**	**23.2373**	**25.7423**	**26.5471**	**24.6379**	17.4776	**25.2369**	**24.4837**	**26.1792**
	Method	1	2	3	4	5	6	7	8	9	10	11
Case (b)	Blurred	21.6800	24.5071	24.0912	21.9366	23.5426	24.1519	23.1112	17.4307	23.6457	23.3993	24.8902
	Levin et al. [2007]	25.5349	30.3812	29.3241	24.2647	27.5837	28.6523	26.4336	17.8144	27.2379	26.4546	28.6267
	Krishnan et al. [2009]	25.6175	30.4072	29.4425	24.3384	27.5730	28.6867	26.4346	17.8007	27.2834	26.4274	28.4925
	IDD-BM3D [2012]	26.4792	**32.1151**	30.0156	25.0356	28.1461	29.4461	26.7580	17.8489	27.6420	26.7408	**28.9194**
	MLP [2013]	26.6398	31.5943	30.1033	**25.1391**	28.3716	29.6208	**26.9048**	**17.8866**	**27.7148**	**26.8471**	28.8552
	Ours	**26.7439**	32.0813	**30.2243**	25.1202	**28.4192**	**29.6550**	**26.9002**	17.8455	**27.7121**	26.8100	28.9167
	Method	1	2	3	4	5	6	7	8	9	10	11
Case (c)	Blurred	18.4181	20.3912	19.7602	19.1161	19.8073	19.9180	19.6269	16.6636	20.0568	20.2844	21.1414
	Levin et al. [2007]	20.8330	22.9116	22.9228	21.0982	22.2095	22.8548	22.0417	17.0863	22.3855	22.0516	23.1149
	Krishnan et al. [2009]	20.9051	22.8331	22.9553	21.1030	22.2017	22.8760	22.0736	17.0853	22.3858	22.0320	23.0646
	IDD-BM3D [2012]	21.3769	**23.9549**	23.5163	21.4466	22.5074	23.2325	22.0854	17.0706	22.6135	22.0592	22.9649
	MLP [2013]	21.6310	23.7254	23.5725	21.5824	22.8560	23.4280	22.2240	**17.0649**	22.7725	22.0889	23.1754
	Ours	**21.6872**	23.8816	**23.6601**	**21.6353**	**22.8711**	**23.4863**	**22.2813**	**17.0651**	**22.7894**	**22.1232**	**23.1865**
	Method	1	2	3	4	5	6	7	8	9	10	11
Case (d)	Blurred	16.8031	18.9516	16.9155	17.3676	17.5742	17.2539	17.9952	16.6595	18.0498	18.9830	20.0871
	Levin et al. [2007]	21.5498	24.2228	22.7576	21.8379	22.3233	22.3097	22.8016	17.3923	22.4392	23.0497	23.8854
	Krishnan et al. [2009]	21.5314	23.9764	22.5647	21.7492	22.1105	22.0626	22.6357	17.3962	22.3222	22.9274	23.7383
	IDD-BM3D [2012]	22.4549	**25.9946**	23.3068	22.5330	23.0620	22.8314	22.9284	**17.5784**	22.6407	23.1095	23.8031
	MLP [2013]	22.3413	25.4975	23.3611	22.7092	23.1639	22.9836	23.0197	17.5717	22.7843	23.0835	23.9968
	Ours	**22.4753**	25.8235	**23.5300**	**22.8066**	**23.2219**	**23.0827**	**23.1123**	17.5587	**22.8476**	**23.2187**	**24.1608**
	Method	1	2	3	4	5	6	7	8	9	10	11
Case (e)	Blurred	16.5474	18.3706	17.0932	17.3106	18.3226	17.5564	17.3743	16.3046	17.8504	18.5884	19.5888
	Levin et al. [2007]	28.8203	31.1460	30.5573	29.0254	29.3529	29.7022	28.2596	19.9890	28.6898	27.9337	29.1433
	Krishnan et al. [2009]	28.8109	30.8978	30.2445	29.6337	29.0727	29.1271	28.0234	20.0545	28.4969	27.6901	28.7134
	IDD-BM3D [2012]	29.2354	**33.3616**	31.2619	31.3124	30.0871	30.6419	28.8559	22.5893	28.9051	28.5393	**29.6615**
	MLP [2013]	29.7103	32.1443	31.0668	31.4754	30.0463	30.5153	28.8545	**23.1308**	28.7937	28.4798	29.4480
	Ours	**30.1910**	32.9455	**31.4396**	**32.5486**	**30.2806**	**30.7379**	**29.0791**	23.1161	**28.9991**	**28.6535**	29.6015

Table 5: *We compare our method to four other state-of-the-art methods. PSNR (in dB) is given for each image in the dataset, for all 5 cases. The average numbers are given in the main paper.*

5 Interlaced HDR

Table 6 lists detailed PSNR numbers of our interlaced HDR (iHDR) reconstructions for different denoisers and different images. The images are shown in Figure 7.

Image	1	2	3	4	5	6	7	8
Missing data	38%	26%	25%	21%	31%	27%	22%	22%
BM3D	46.51	36.29	45.98	46.49	44.22	42.06	40.69	42.17
NLM	39.79	34.14	45.22	45.03	42.28	39.63	39.29	40.22
Patchwise NLM	46.70	35.59	45.81	46.41	43.90	41.83	40.96	42.20
Sliding DCT	43.03	30.30	44.67	42.75	41.59	29.95	39.27	38.03
Averaging	46.26	35.97	45.43	46.08	43.79	42.10	39.52	41.79

Table 6: *PSNR values for different interlaced HDR images using different denoisers.*

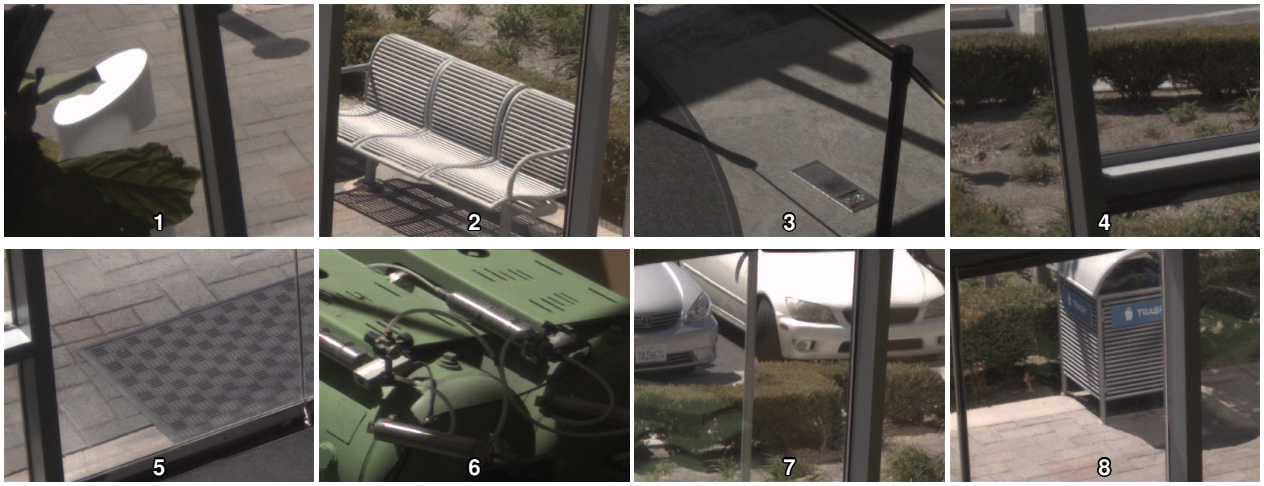


Figure 7: *Images used for the interlaced HDR reconstruction evaluation.*

5.1 User-defined exposure weight mask

Although by default the matrix **M** discards only useless pixel data, it can be also used to bias the solution towards the short- or long-exposure parts. We demonstrate this in Fig. 8, where the user marked the surfer and limited the reconstruction of the underlying image to only use the short-exposure data, to prefer a sharper, less blurry (but also noisier) result. The selection mask can be created manually, as in Fig. 8 (b), or with a stroke-based edit propagation framework, such as the one introduced by Baek et al. [2013].

Figure 8: *Image reconstruction with user-defined exposure weight mask.* ***(a)*** *Sparse observations; many short-exposure pixels are too dark to be useful, and the long-exposure data has a lot of motion-induced blur.* ***(b)*** *Sparse observations with user-defined (short-)exposure preference mask.* ***(c)*** *The output of our method without the use of the mask. The surfer and details on his board have little noise, but are both affected by motion-blur.* ***(d)*** *With the exposure weight mask applied, we can trade-off motion-related artifacts for increased amount of detail (and noise).*

6 Color Camera Array

Figure 9 shows a number of scenes that were captured with the (small baseline) 2×2 prototype sensor. Note that there are no visible color artifacts in the images. Note also that the lenses on this prototype sensor are not ideal, and the input images are somewhat soft (see also the supplemental web page).

We note that our method can directly, and almost trivially handle stuck pixel masking, instead of relying on a separate part of the pipeline (where demosaicking is typically not aware that stuck pixels were filled with averaged data). An example with 2×2 color array camera input is shown in Figure 10, the same approach works just as well for Bayer images.

(a) Red channel **(b)** Green channel **(c)** Blue channel **(d)** Our reconstruction

Figure 9: *A few 2×2 example datasets reconstructed with our method. Note how there are no visible color artifacts in the results (all examples use the cross-channel prior).*

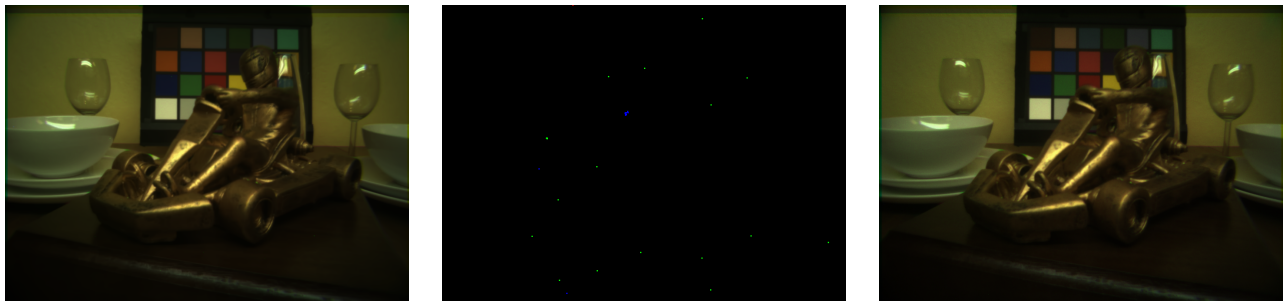


Figure 10: *Stuck pixel masking can be directly incorporated into our method as well. Here we show a result image without stuck pixel masking (left), a (conservative) mask indicating stuck pixels (middle), and our result (right). Please zoom in to see the differences.*

6.1 4 × 4 Camera Array

In addition to the 2×2 camera array cameras, we have experimented with a simulated 4×4 array. For this we used the Stanford light field dataset [Stanford Graphics Laboratory 2008] by picking views and dropping color channels in the same pattern as in PiCam [Venkataraman et al. 2013]. Each view is first blurred using a 2×2 box kernel and then decimated to simulate loss in resolution. Using this simulated 4×4 array, we aim to achieve $2\times$ superresolved (w.r.t. a single tile) images, but with consistent colors. We follow the standard warp-blur-decimate forward model for superresolution [Mitzel et al. 2009; Liu and Sun 2011] to recover some of the lost resolution by utilizing subpixel shifts between the different color channel captures. We set $\mathbf{A}_i = \mathbf{M}_i\mathbf{DBS}_i$ for each capture i (out of k), and get

$$G(\mathbf{x}) = \|[\mathbf{M}_1\mathbf{z}_1;\ldots;\mathbf{M}_k\mathbf{z}_k] - [\mathbf{M}_1\mathbf{DBS}_1;\ldots;\mathbf{M}_k\mathbf{DBS}_k]\mathbf{x}\|_2^2. \tag{3}$$

The blur matrix $\mathbf{B}$ represents convolution with a 2×2 box kernel and models 100% fill-factor square pixels without any anti-aliasing filter on top of the sensor, and the decimation matrix $\mathbf{D}$ represents the uniform $2\times$ downsampling. To get a baseline, we first computed optical flow between the tiles (see Section 4.4 in the main paper), and applied bicubic upsampling both to the flow and colors into a $2\times$ denser grid. The data was forward-warped, and normalized to create the output image (Fig. 11 left). We then ran our method, which first computes the optical flow as before and then used our framework to get the results in the center column of Fig. 11.

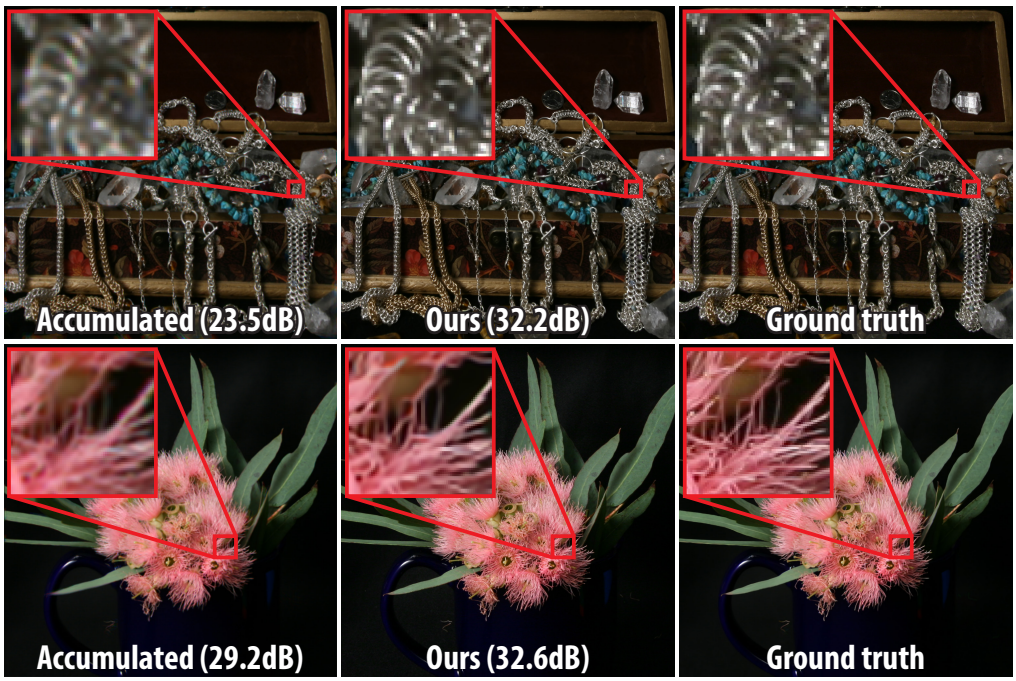
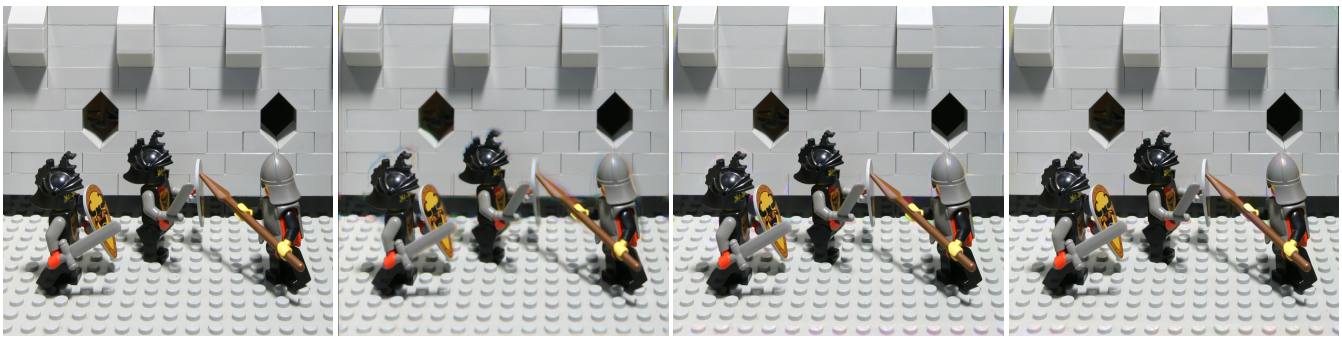


Figure 11: *Superresolution with light field data simulating a 4×4 array camera. (**left**) Straightforward baseline image that upsamples the images and warps before accumulating into a $2\times$ finer grid. (**middle**) Our algorithm effectively combines natural image priors for performing a faithful superresolution. (**right**) Ground truth.*

Figure 12 shows how the use of the cross-channel prior removes potential color artifacts. The artifacts are very evident in (b) and (c), whereas our final result in (d) shows very few artifacts.

6.2 Iterated Application

We can apply Alg. 1 iteratively to improve the quality of the superresolved RGB images. At the first iteration, we use the cross-channel scaled gradients to initialize the flow. Reconstruction after this first iteration can show banding artifacts resulting from small alignment inaccuracies, which get enhanced by the 2×2 deconvolution. In subsequent iterations (Fig. 13), we use the reconstructed superresolved image as the reference for motion estimation, and compute the red channel flow to the red channel of the current output, and the same for the blue channel. This achieves a better alignment and results in no banding. Note that any artifacts are not due to demosaicking, as the sensors do not use the Bayer color pattern.



(a) Original **(b)** Accumulate **(c)** Ours (**without** cross-channel prior) **(d)** Ours (**with** cross-channel prior)

Figure 12: *Another* 4×4 *simulated dataset reconstructed with our method. Note how there are many color artifacts (e.g., around the left helmet), when the cross-channel prior is not used. The cross-channel prior removes almost all artifacts. Please zoom into the digital version of the supplemental for comparison.*

Figure 13: *Due to imprecise initial cross-channel optical flow, the first result has chromatic artifacts. Iterating between reconstruction and flow estimation improves both the flow and the reconstruction.*

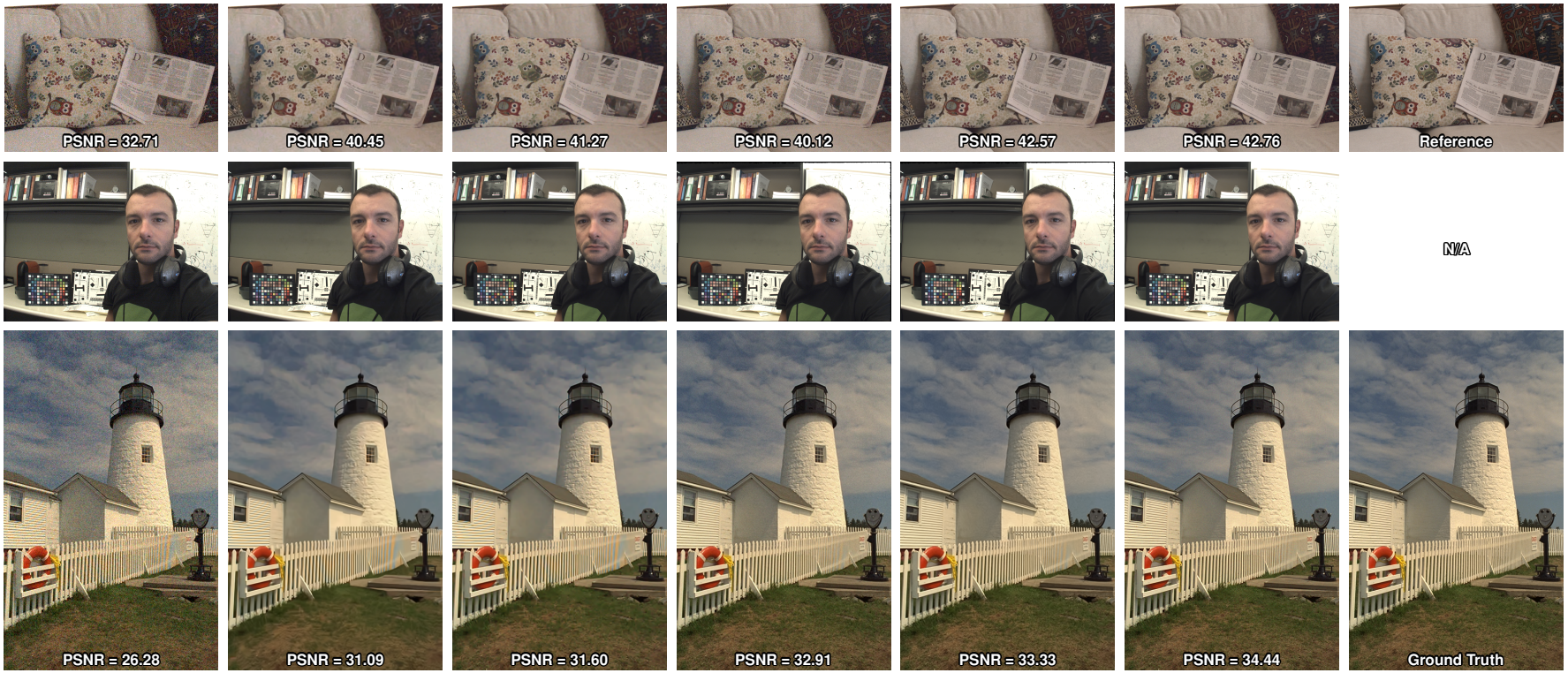
7 Burst Denoising and Demosaicking

We include two real-world examples and one simulated example of burst denoising and demosaicking in Figure 15. Note that all the examples can also be browsed and compared in the supplemental web page, which provides a more intuitive interface for comparisons.

In Figure 14 we show that there are often two sets of parameters leading to high PSNR, but with different visual appeal. In this case (a) appears very crisp, but has also slightly more noise, whereas (b) is virtually noise-free, but appears slightly softer. Which result is preferred is a matter of personal choice. In the paper, we always include the result with the highest PSNR.

(a) Ours – Crisp **(b)** Ours – Noise Free **(c)** Ground truth

Figure 14: *The tables in the paper report the results with the highest PSNR. However, a visually different result with very similar PSNR may be achievable by adjusting the parameters. In this case, the image in (a) appears crisp but contains noise, whereas (b) appears noise-free, but just a little bit softer. Both have virtually the same PSNR. Image (b) was created by slightly increasing the strength of the denoising prior. (Image courtesy of Flickr user susan402.)*

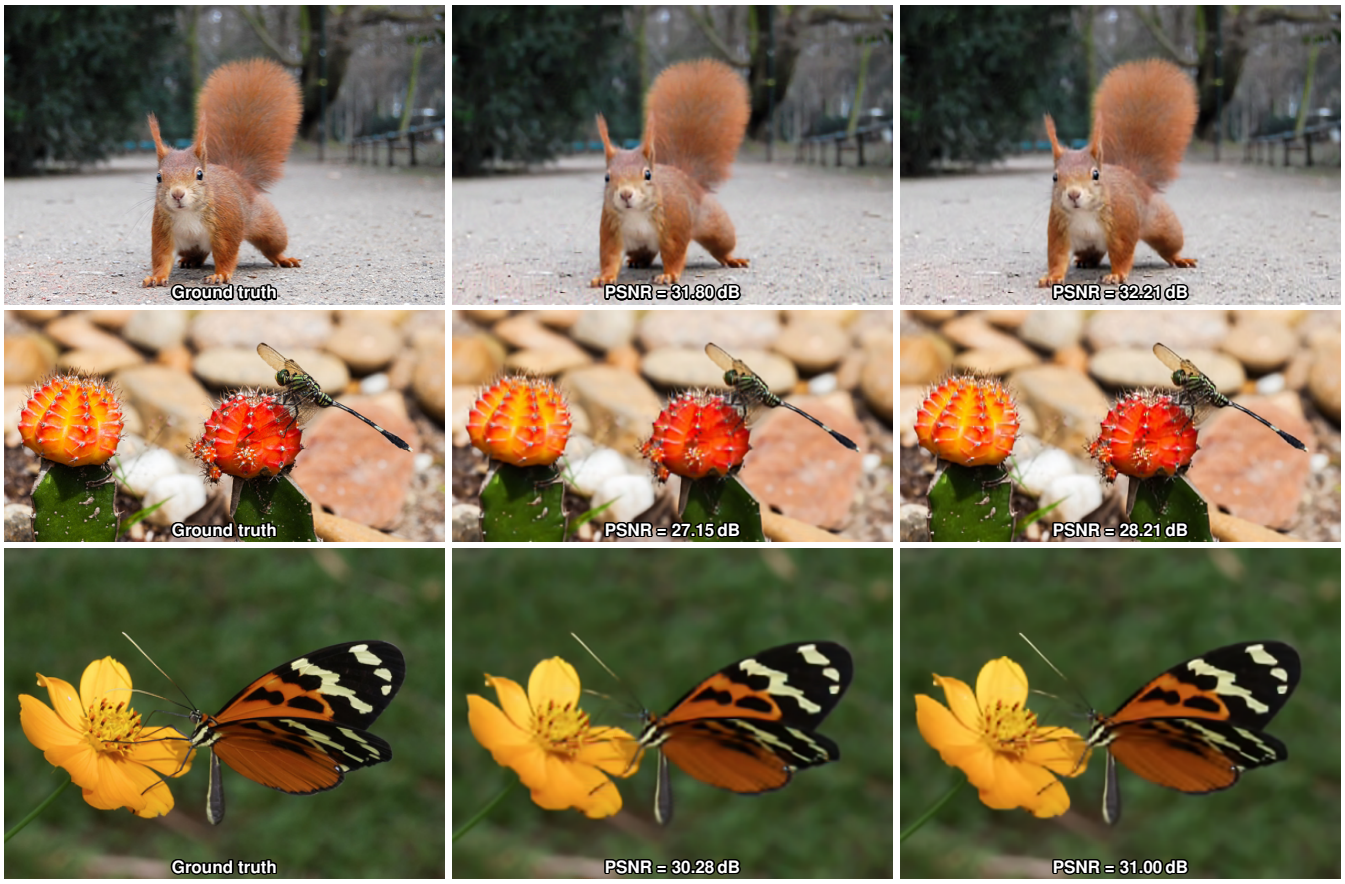


(a) First frame of stack **(b)** BM3D (first frame) **(c)** VBM3D (stack) **(d)** NLM-weighted stack **(e)** NLM stack + BM3D **(f)** Ours **(g)** Reference

Figure 15: *Burst denoising and demosaicking—additional examples. The top row shows a real-world example; the stack of 8 images was captured at ISO 12800 with a Canon EOS 650D, except for the reference image (captured at ISO 100). We note that the captured reference image cannot be considered reliable "ground truth", as demosaicking is required. We encourage the reader to do a qualitative comparison using the supplemental web page. The second row shows a handheld portrait captured at 30fps on a 3.2MP PointGrey Flea 3, yielding a stack of 16 noisy images. Since the scene was dynamic, it was not possible to capture a reference image, and no PSNR can be given. Again, we encourage the reader to check our supplemental web page. The third dataset, showing the famous Cape Neddick Lighthouse, is a simulated dataset with additive white Gaussian noise ($\sigma = 12/255$) applied to each of the 16 images in the dataset. Our quality improvement can be clearly seen.*

8 Beyond RGB

We demonstrate a few examples images in Figure 16, where we have directly reconstructed an image in YUV420 from an (simulated) interlaced HDR image. We compare this against the ground truth image, and the pipelined approach, where we first reconstruct to RGB and then simply convert to YUV420. Our joint reconstruction yields a visible improvement. Table 7 shows the PSNR numbers for all twelve images that were used in our experiment.



(a) Ground truth **(b)** Pipeline **(c)** Ours

Figure 16: *Visual comparison between first reconstructing an interlaced HDR image as RGB and then converting to YUV420 (pipeline approach) and directly reconstructing into YUV420 using our framework. (Images courtesy Wikemedia users Ray Eye, Diego Delso, and Hans Hillewaert.)*

	Biandintz	Church	Eichhorn	Gull	Houses	Cow	Rally	Yundrok	Libelle	Melinaea	Mototaxis	Platycercus
Pipeline	29.12	29.26	31.80	34.57	26.12	26.10	**26.03**	**26.74**	27.15	30.28	28.52	25.67
Joint (Ours)	**29.36**	**29.71**	**32.21**	**35.31**	**26.15**	**26.78**	**26.03**	26.60	**28.21**	**31.00**	**28.86**	**25.74**

Table 7: *Comparison between first reconstructing an interlaced HDR image as RGB and then converting to YUV420 (pipeline approach) and directly reconstructing into YUV420 using our framework (PSNR values in dB).*

9 Deblocking

In Table 8 we compare our method for reconstructing JPEG compressed images to the state-of-the-art shape-adaptive DCT (SA-DCT) deblocker [Foi et al. 2007].

Quality Q	Lena			Peppers			F-16			Baboon			Lake			Tiffany			House			Avg		
	JPEG	SA-DCT	Ours	JPEG	SA-DCT	Ours	JPEG	SA-DCT	Ours	JPEG	SA-DCT	Ours	JPEG	SA-DCT	Ours	JPEG	SA-DCT	Ours	JPEG	SA-DCT	Ours	JPEG	SA-DCT	Ours
75	33.21	33.56	**33.75**	30.29	**30.67**	30.65	32.61	33.18	**33.29**	26.21	26.25	**26.51**	28.65	28.91	**28.99**	31.03	31.19	**31.50**	31.44	**32.00**	31.96	30.49	30.82	**30.95**
50	32.02	32.63	**32.81**	29.25	29.81	**29.82**	31.06	31.83	**31.95**	24.85	24.97	**25.25**	27.66	28.07	**28.13**	29.91	30.12	**30.44**	29.80	**30.40**	30.39	29.22	29.69	**29.83**
40	31.54	32.26	**32.44**	28.83	29.45	**29.46**	30.64	31.52	**31.59**	24.40	24.56	**24.82**	27.31	27.78	**27.84**	29.57	29.87	**30.12**	29.51	**30.20**	30.17	28.83	29.38	**29.49**
30	30.91	31.79	**31.93**	28.40	**29.14**	29.12	30.06	31.09	**31.15**	23.85	24.06	**24.29**	26.84	27.38	**27.44**	29.21	29.62	**29.83**	28.96	**29.76**	29.73	28.32	28.98	**29.07**
25	30.44	31.46	**31.55**	28.04	**28.90**	28.85	29.58	30.71	**30.73**	23.50	23.75	**23.95**	26.51	27.13	**27.18**	28.91	29.40	**29.60**	28.55	**29.44**	29.38	27.93	28.68	**28.75**
20	29.83	31.00	**31.02**	27.57	**28.53**	28.46	28.90	**30.13**	30.09	23.07	23.37	**23.53**	26.07	26.78	**26.80**	28.40	28.99	**29.12**	27.87	**28.75**	28.68	27.39	28.22	**28.24**

Table 8: *PSNR value (in dB) for different JPEG quality metrics. The bold numbers indicate the best method.*

References

BAEK, J., PAJĄK, D., KIM, K., PULLI, K., AND LEVOY, M. 2013. Wysiwyg computational photography via viewfinder editing. *ACM Trans. Graph. 32*, 6.

DANIELYAN, A., KATKOVNIK, V., AND EGIAZARIAN, K. 2012. BM3D frames and variational image deblurring. *IEEE TIP 21*, 4.

FOI, A., KATKOVNIK, V., AND EGIAZARIAN, K. 2007. Pointwise shape-adaptive DCT for high-quality denoising and deblocking of grayscale and color images. *IEEE TIP 16*, 5.

JEON, G., AND DUBOIS, E. 2013. Demosaicking of noisy Bayer-sampled color images with least-squares luma-chroma demultiplexing and noise level estimation. *IEEE TIP 22*, 1.

KRISHNAN, D., AND FERGUS, R. 2009. Fast image deconvolution using hyper-laplacian priors. In *NIPS*.

LEVIN, A., FERGUS, R., DURAND, F., AND FREEMAN, W. T. 2007. Deconvolution using natural image priors. *ACM Transactions on Graphics (TOG) 26*, 3.

LEVIN, A., WEISS, Y., DURAND, F., AND FREEMAN, W. T. 2009. Understanding and evaluating blind deconvolution algorithms. In *IEEE CVPR*.

LIU, C., AND SUN, D. 2011. A Bayesian approach to adaptive video super resolution. In *CVPR*.

MALVAR, H. S., HE, L.-W., AND CUTLER, R. 2004. High-quality linear interpolation for demosaicing of bayer-patterned color images. In *ICASS*.

MITZEL, D., POCK, T., SCHOENEMANN, T., AND CREMERS, D. 2009. Video super resolution using duality based TV-L_1 optical flow. In *Pattern Recognition*. Springer.

SCHULER, C. J., BURGER, H. C., HARMELING, S., AND SCHÖLKOPF, B. 2013. A machine learning approach for non-blind image deconvolution. In *CVPR*.

STANFORD GRAPHICS LABORATORY, 2008. The (new) stanford light field archive. `http://lightfield.stanford.edu/`.

VENKATARAMAN, K., LELESCU, D., DUPARRÉ, J., MCMAHON, A., MOLINA, G., CHATTERJEE, P., MULLIS, R., AND NAYAR, S. 2013. Picam: an ultra-thin high performance monolithic camera array. *ACM TOG 32*, 6.

XU, L., AND JIA, J. 2010. Two-phase kernel estimation for robust motion deblurring. In *ECCV*.

ZHANG, L., WU, X., BUADES, A., AND LI, X. 2011. Color demosaicking by local directional interpolation and nonlocal adaptive thresholding. *Journal of Electronic Imaging 20*, 2.