

Supplementary Document

In this document, we present additional results and implementation details regarding our statistical error reduction framework for Monte Carlo (MC) rendering. The source code and other materials are available at <https://www.cg.tuwien.ac.at/StatER>.

In Alg. 1, we first summarize how to compute online statistics, with which we augment the existing physically based renderer pbrt-v3. We specify our novel multi-transform variance denoising approach in Alg. 2. The resulting low-error variance estimates can then be used for cascaded denoising of the image (i.e., per-pixel mean estimates) itself in Alg. 3 (which also involves multiple transforms). Finally, we utilize our denoised variance estimates for iterative adaptive sampling as described in Alg. 4.

In the following sections, we present further experimental results demonstrating the performance of our approach in various configurations. Furthermore, Figs. S3–S18 showcase additional results and compare image error metrics for our denoising and adaptive sampling framework to relevant related methods. In Table S1, we provide denoising times for our main results; timings include GPU up- and download. Due to implementation details, ASM+MCL estimates variance twice; removing this redundancy could notably reduce the reported timings.

S1 Choosing the Number of Transformations

As stated in the paper, the improvement in quality from one to two transformations is notable, but further transformations provide diminishing returns. When comparing methods under equal-time conditions, these marginal gains are outweighed by the overhead of tracking and utilizing the extra statistics for denoising. Therefore, we use $\{\lambda_m\} = \{0.25, 1\}$ throughout the paper. In Fig. S1, we provide equal-time plots illustrating this effect for the case of adaptive sampling, as well as a visual comparison in Fig. S16. Both also include a case (“MT3-JB”) for using the Jarque–Bera statistic instead of the default most-discriminative heuristic for mean denoising.

S2 Combining Adaptive Sampling and Denoising

When combining adaptive sampling (§5.3.1) and denoising (§4), further variance reduction can be achieved by weighting pixel estimates by their sample sizes¹. Specifically, for mean denoising, we adapt Eq. (2) to $w_{ij} = \rho_{ij}m_{ij} \min(n_i, n_j) / (\sum_i \rho_{ij}m_{ij} \min(n_i, n_j))$, where $\min(n_i, n_j)$ prevents increasing bias from estimates $\hat{\theta}_i$ where $n_i \gg n_j$ and $m_{ij} = 1$. For variance denoising, where we prefer noise reduction over bias avoidance (see §5.1), we use (Bessel-corrected) degrees of freedom $(n_i - 1)$ instead of $\min(n_i, n_j)$. These adaptations are included in Algs. 2–3.

S3 Sampler Robustness

We evaluated the robustness of our statistical denoiser by testing a publicly available implementation of the Z sampler [Ahmed and Wonka 2020], known for its blue-noise characteristics. As shown in Fig. S2, the denoising quality is comparable to that obtained with the random sampler (used throughout our paper), indicating consistent performance across samplers.

¹ William G. Cochran. 1954. The Combination of Estimates from Different Experiments. *Biometrics* 10, 1 (1954), 101–129. doi:10.2307/3001666

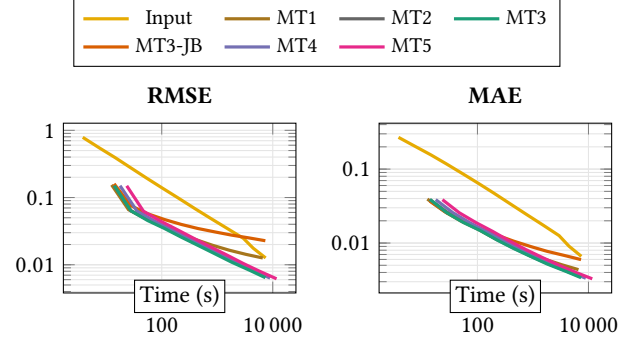


Fig. S1. Plots illustrating the trade-off associated with the used number of transforms for adaptive sampling and denoising (ASM+MCL). Here, we have used the KITCHEN scene (Fig. S10) and the following transformation sets—MT1: $\{0.5\}$, MT2: $\{0.5, 0.75\}$, **MT3** (default): $\{0.25, 1\}$, MT4: $\{0.25, 0.5, 0.75, 1\}$, MT5: $\{0.01, 0.25, 0.5, 0.75, 1, 3\}$, as well as MT3 using the Jarque–Bera statistic instead of the most-discriminative heuristic (MT3-JB).

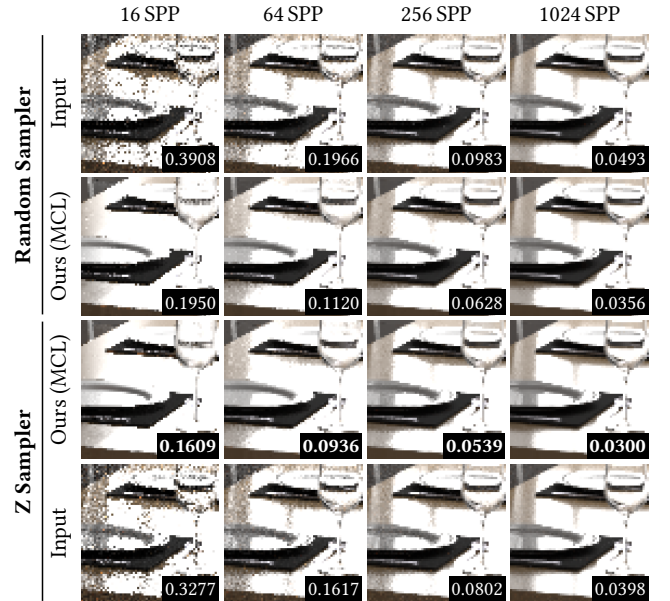


Fig. S2. Comparison of denoising quality using pbrt-v3’s random sampler (used throughout our paper) and the (blue-noise) Z sampler, showing comparable overall performance (errors are reported as RMSE).

Algorithm 1: Online statistics

```

/* After variables set to 0; per sample  $\ell_{ik}$ , per pixel  $i$ 
   with position  $(x, y) \in [0, 1]^2$  during rendering: */
 $n_i \leftarrow n_i + 1$ 
foreach multi-transform  $\lambda_m$  do
    Box-Cox transform  $\ell_{ik} \rightarrow \ell_{ik}^{(\lambda_m)}$  /* Eq. (5) */
     $\delta \leftarrow \ell_{ik}^{(\lambda_m)} - \hat{\theta}_i^{(\lambda_m)}$  /* Moments [Meng 2015] */
     $\hat{\theta}_i^{(\lambda_m)} \leftarrow \hat{\theta}_i^{(\lambda_m)} + \delta/n_i$ 
     $\hat{M}_{2,i}^{(\lambda_m)} \leftarrow \hat{M}_{2,i}^{(\lambda_m)} + \delta(\delta - \delta/n_i)$ 
     $\hat{M}_{3,i}^{(\lambda_m)} \leftarrow \hat{M}_{3,i}^{(\lambda_m)} - 3(\delta/n_i)\hat{M}_{2,i}^{(\lambda_m)} + \delta(\delta^2 - (\delta/n_i)^2)$ 
     $\hat{M}_{4,i}^{(\lambda_m)} \leftarrow \hat{M}_{4,i}^{(\lambda_m)} - 4(\delta/n_i)\hat{M}_{3,i}^{(\lambda_m)} - 6(\delta/n_i)^2\hat{M}_{2,i}^{(\lambda_m)} +$ 
     $\delta(\delta^3 - (\delta/n_i)^3)$ 
     $\hat{C}_{x,i}^{(\lambda_m)} \leftarrow \hat{C}_{x,i}^{(\lambda_m)} + (\ell_{ik}^{(\lambda_m)} - \hat{\theta}_i^{(\lambda_m)})(x - 1/2)$  /* Covs. */
     $\hat{C}_{y,i}^{(\lambda_m)} \leftarrow \hat{C}_{y,i}^{(\lambda_m)} + (\ell_{ik}^{(\lambda_m)} - \hat{\theta}_i^{(\lambda_m)})(y - 1/2)$ 
     $\hat{M}_{1,i}^{(\lambda_m)} \leftarrow \hat{M}_{1,i}^{(\lambda_m)} + \sqrt{\delta(\delta - \delta/n_i)}$  /* Mean abs. dev. */

 $\hat{\mu}_{r,i}^{(\lambda_m)} \leftarrow \hat{M}_{r,i}^{(\lambda_m)}/n_i$  /* Central moments,  $r = \{2, 3, 4\}$  */
 $\hat{\sigma}_i^{2,(\lambda_m)} \leftarrow \hat{M}_{2,i}^{(\lambda_m)}/(n_i - 1)$  /* Bessel-corr. variance */
 $\hat{\Sigma}_{x,i}^{(\lambda_m)} \leftarrow \hat{C}_{x,i}^{(\lambda_m)}/(n_i - 1)$  /* Covariance, x */
 $\hat{\Sigma}_{y,i}^{(\lambda_m)} \leftarrow \hat{C}_{y,i}^{(\lambda_m)}/(n_i - 1)$  /* Covariance, y */
 $\hat{\mu}_{1,i}^{(\lambda_m)} \leftarrow \hat{M}_{1,i}^{(\lambda_m)}/n_i$  /* Mean absolute deviation */

Analogously compute untransformed  $\hat{\theta}_i$  and  $\hat{\sigma}_i^2$ 

```

Algorithm 3: Multi-transform mean denoising

Input: online statistics (Algorithm 1), G-buffer vectors $\mathbf{p}_j$,
potentially LEV-corrected $\hat{\sigma}_j^{2,(\lambda_m)}$ or denoised
variances $\tilde{\sigma}_j^{2,(\lambda_m)}$ (Algorithm 2)

Output: denoised means $\tilde{\theta}_j$

```

foreach center pixel  $j$  in parallel do
     $\{W, \tilde{\theta}_j\} \leftarrow 0$ 
    foreach candidate pixel  $i$  within filter radius do
        Evaluate base filter  $(\mathbf{p}_i, \mathbf{p}_j) \rightarrow \rho_{ij}$  /* Eq. (21) */
        foreach multi-transform  $\lambda_m$  do
             $\tilde{\sigma}_i^2 \leftarrow \hat{\sigma}_i^{2,(\lambda_m)}$ ;  $\tilde{\sigma}_j^2 \leftarrow \hat{\sigma}_j^{2,(\lambda_m)}$ 
            if cascaded denoising then
                 $\tilde{\sigma}_i^2 \leftarrow \max(\hat{\sigma}_i^{2,(\lambda_m)}, \tilde{\sigma}_i^{2,(\lambda_m)})$ 
                 $\tilde{\sigma}_j^2 \leftarrow \max(\hat{\sigma}_j^{2,(\lambda_m)}, \tilde{\sigma}_j^{2,(\lambda_m)})$ 
                 $t \leftarrow (\hat{\theta}_j^{(\lambda_m)} - \hat{\theta}_i^{(\lambda_m)})/(\tilde{\sigma}_i^2/n_i + \tilde{\sigma}_j^2/n_j)^{\frac{1}{2}}$ 
                 $m_{ij}^{(\lambda_m)} \leftarrow |t| < t_{v,1-\alpha/2} ? 1 : 0$  /* Eq. (4) */
            /* Use most discriminative  $\lambda_m$ : */
             $w_{ij} \leftarrow \rho_{ij} \min_{\lambda_m} m_{ij}^{(\lambda_m)}$  (adapt. sampling?  $\min(n_i, n_j) : 1$ )
             $W \leftarrow W + w_{ij}$ 
             $\tilde{\theta}_j \leftarrow \tilde{\theta}_j + w_{ij}\hat{\theta}_i$ 
         $\tilde{\theta}_j \leftarrow \tilde{\theta}_j/W$ 

```

Algorithm 2: Multi-transform variance denoising

Input: online statistics (Algorithm 1), G-buffer vectors $\mathbf{p}_j$
Output: potentially LEV-corrected $\hat{\sigma}_j^{2,(\lambda_m)}$ or denoised
variances $\tilde{\sigma}_j^{2,(\lambda_m)}, \tilde{\sigma}_j^2$

```

if linear explained-variance correction then
    foreach pixel  $j$  in parallel do
        foreach multi-transform  $\lambda_m$  do
             $\hat{\sigma}_j^{2,(\lambda_m)} \leftarrow \hat{\sigma}_j^{2,(\lambda_m)} - 4\hat{\mu}_{1,j}^{(\lambda_m)} (|\hat{\Sigma}_{x,j}^{(\lambda_m)}| + |\hat{\Sigma}_{y,j}^{(\lambda_m)}|)$  /* Eq. (17) */

foreach center pixel  $j$  in parallel do
     $\{W, \tilde{\sigma}_j^{2,(\lambda_m)}, \tilde{\sigma}_j^2\} \leftarrow 0$ 
    foreach candidate pixel  $i$  within filter radius do
        Evaluate base filter  $(\mathbf{p}_i, \mathbf{p}_j) \rightarrow \rho_{ij}$  /* Eq. (21) */
         $\lambda_m^* \leftarrow \arg \min_{\lambda_m} \text{JB}_{ij}^{(\lambda_m)}$  /* best  $\lambda_m$ , Eq. (10) */
        /* Evaluate Eq. (18) using statistics of  $\lambda_m^*$ -transformed samples: */
         $m_{ij} \leftarrow \hat{\sigma}_{\Delta}^{2,(\lambda_m^*)} \leq \hat{s}\hat{e}_{\Delta}^{(\lambda_m^*)} ? 1 : 0$ 
         $w_{ij} \leftarrow \rho_{ij}m_{ij}$  (adaptive sampling?  $(n_i - 1) : 1$ )
         $W \leftarrow W + w_{ij}$ 
        if cascaded denoising then
            foreach multi-transform  $\lambda_m$  do
                 $\tilde{\sigma}_j^{2,(\lambda_m)} \leftarrow \tilde{\sigma}_j^{2,(\lambda_m)} + w_{ij}\hat{\sigma}_i^{2,(\lambda_m)}$ 
             $\tilde{\sigma}_j^2 \leftarrow \tilde{\sigma}_j^2 + w_{ij}\hat{\sigma}_i^2$ 
        if cascaded denoising then
            foreach multi-transform  $\lambda_m$  do
                 $\tilde{\sigma}_j^{2,(\lambda_m)} \leftarrow \tilde{\sigma}_j^{2,(\lambda_m)}/W$ 
             $\tilde{\sigma}_j^2 \leftarrow \tilde{\sigma}_j^2/W$ 

```

Algorithm 4: Iterative optimal adaptive sampling

Input: noisy $\hat{\sigma}_j^2$ and denoised $\tilde{\sigma}_j^2$ variances (Algorithm 2;
both uncorrected), number of optimization steps S
(default: 8)

Output: optimal SPPs $n_{j,l+1}^*$ for iteration $l + 1$

```

/* Set total sample budget  $n$  to the past  $\sum_j n_j$  and
   remaining total number of samples  $B_{l+1}$  for the next
   rendering iteration  $l + 1$ : */
 $n \leftarrow \sum_j n_j + B_{l+1}$ 
 $\forall j : \tilde{\sigma}_j \leftarrow \max(\hat{\sigma}_j, \tilde{\sigma}_j)$ 
for  $s \leftarrow 1$  to  $S$  do
    foreach pixel  $j$  in parallel do
        /* Calculate number of samples for pixel  $j$  for
           next iteration  $l + 1$ ; take at least 4: */
         $n_{j,l+1} \leftarrow n\tilde{\sigma}_j/\sum_k \tilde{\sigma}_k - n_j$  /* float */
         $n_{j,l+1}^* \leftarrow \max(\text{round}(n_{j,l+1}), 4)$  /* integer */
    /* Update total sample budget  $n$ , correcting for
       max() and round(): */
     $n \leftarrow \sum_j (n_j + n_{j,l+1} - n_{j,l+1}^*)$ 

```

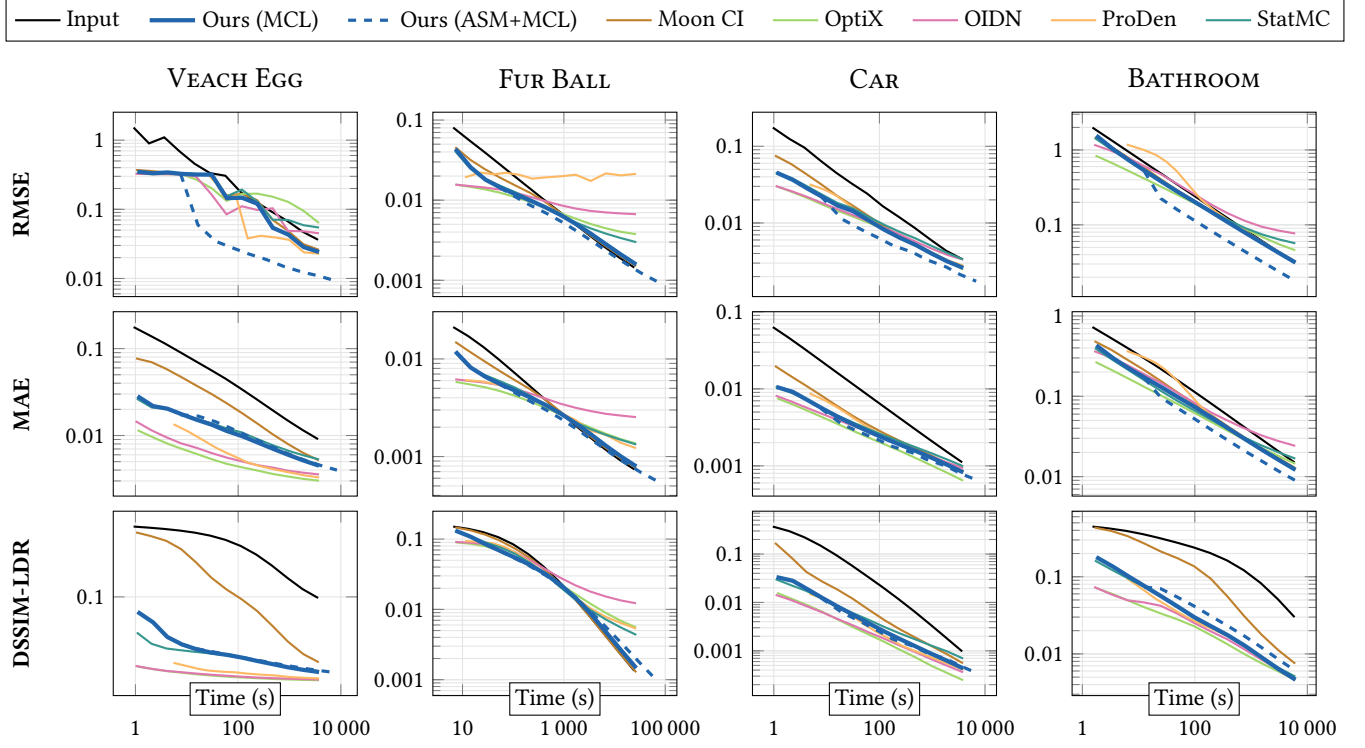


Fig. S3. Equal-time comparison for the results presented in our paper (Fig. 15).

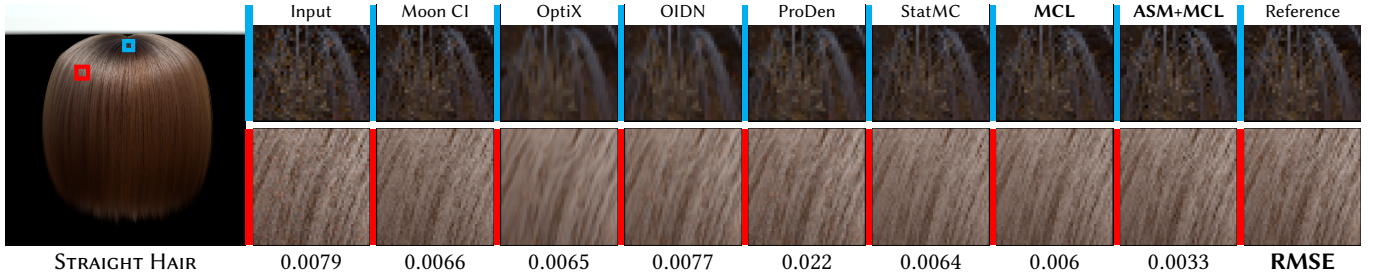


Fig. S4. STRAIGHT HAIR rendered with 512 SPP (average for ASM+MCL).

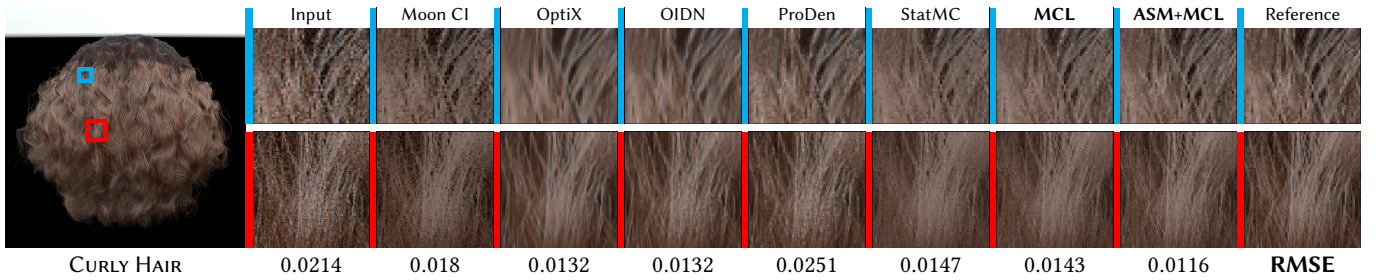


Fig. S5. CURLY HAIR rendered with 128 SPP (average for ASM+MCL).

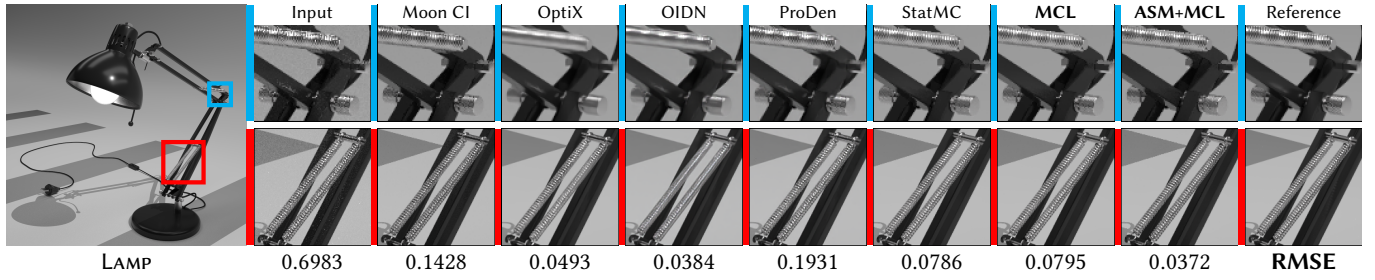


Fig. S6. LAMP rendered with 128 SPP (average for ASM+MCL).

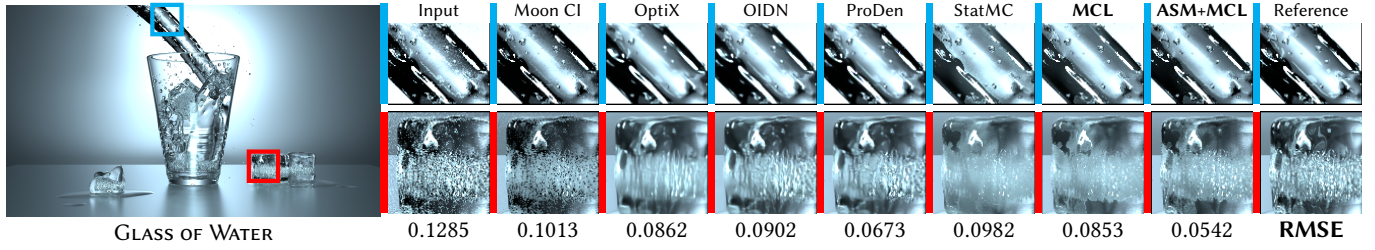


Fig. S7. GLASS OF WATER rendered with 128 SPP (average for ASM+MCL).

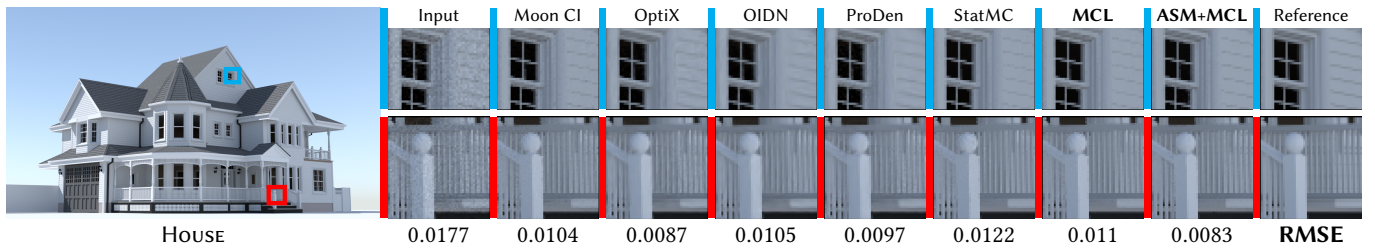


Fig. S8. HOUSE rendered with 256 SPP (average for ASM+MCL).

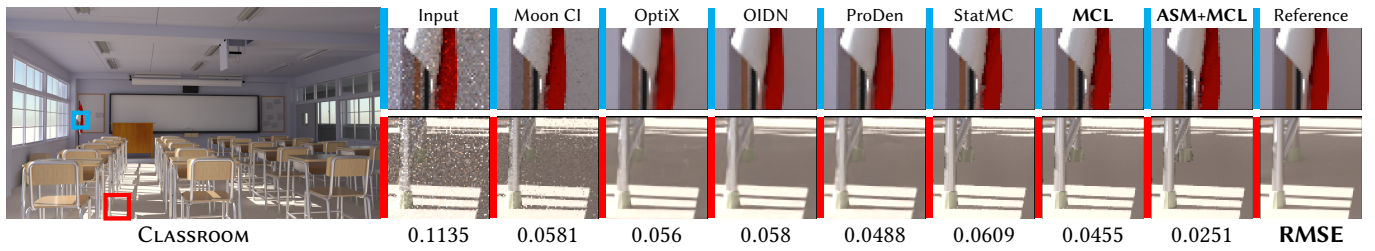


Fig. S9. CLASSROOM rendered with 128 SPP (average for ASM+MCL).

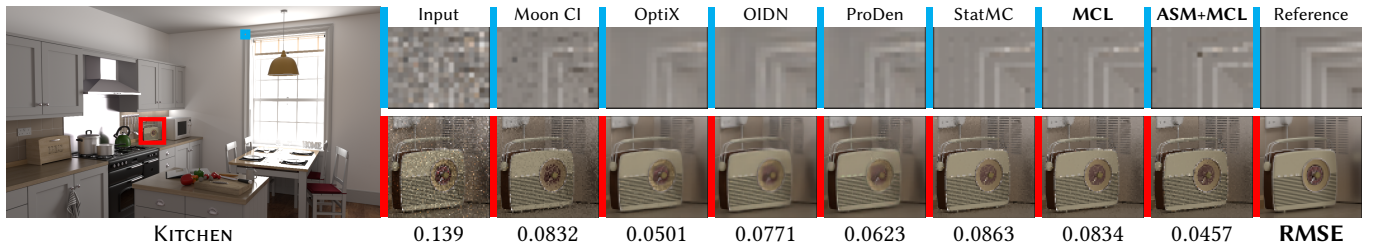


Fig. S10. KITCHEN rendered with 128 SPP (average for ASM+MCL).

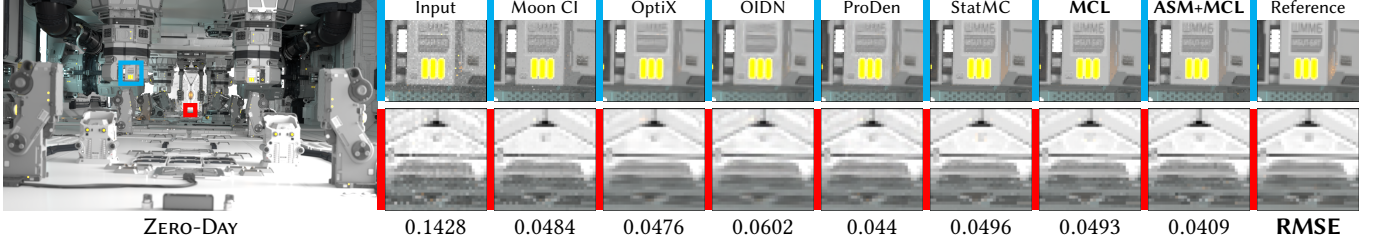


Fig. S11. ZERO-DAY rendered with 512 SPP (average for ASM+MCL).

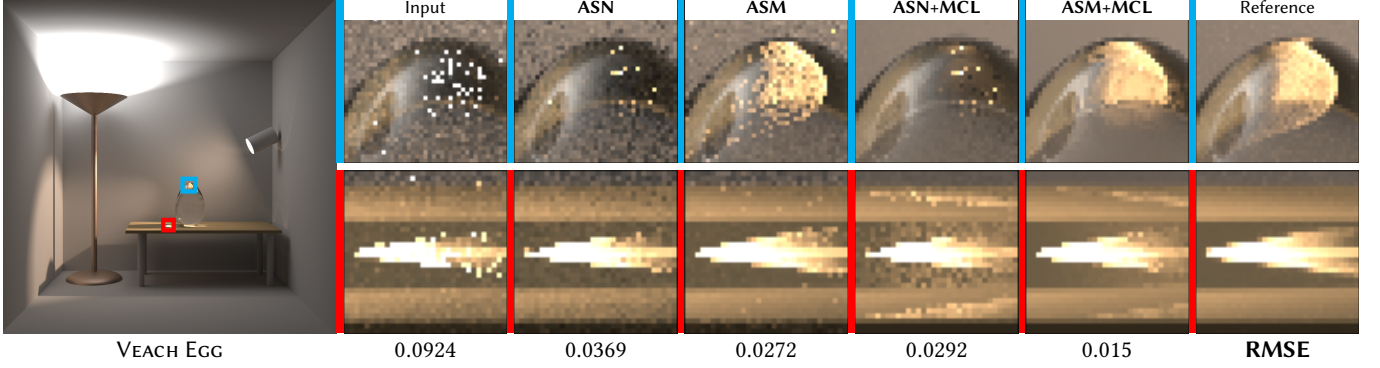


Fig. S12. Comparison of adaptive sampling with (ASM) and without (ASN) multi-transform variance denoising, before and after mean denoising (MCL), for VEACH EGG rendered with 2048 SPP (average for ASN and ASM).

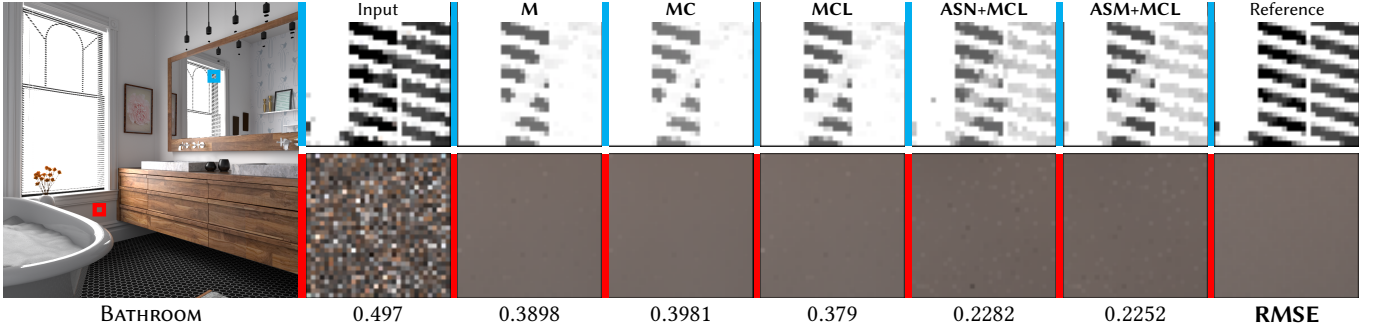


Fig. S13. Ablation study on our denoising components for BATHROOM rendered with 64 SPP (average for ASN and ASM): multi-transform denoising (M), cascaded denoising (C), linear explained-variance correction (L), and adaptive sampling with (ASM) and without (ASN) multi-transform variance denoising.

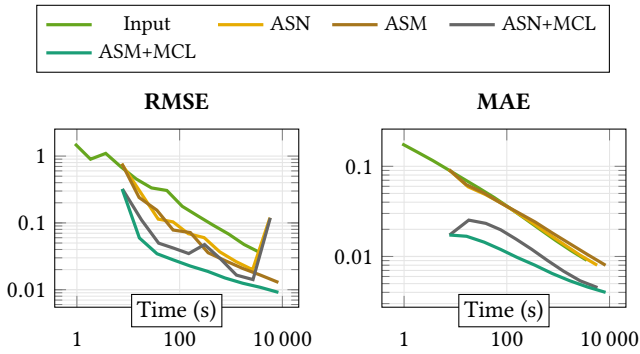


Fig. S14. Image-error plots for the results in Fig. S12 (VEACH EGG).

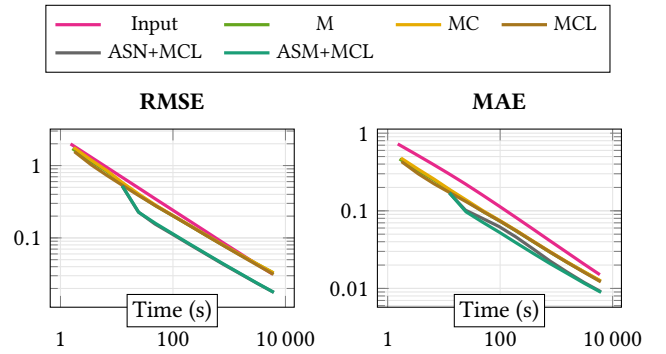


Fig. S15. Image-error plots for the ablation study in Fig. S13 (BATHROOM).

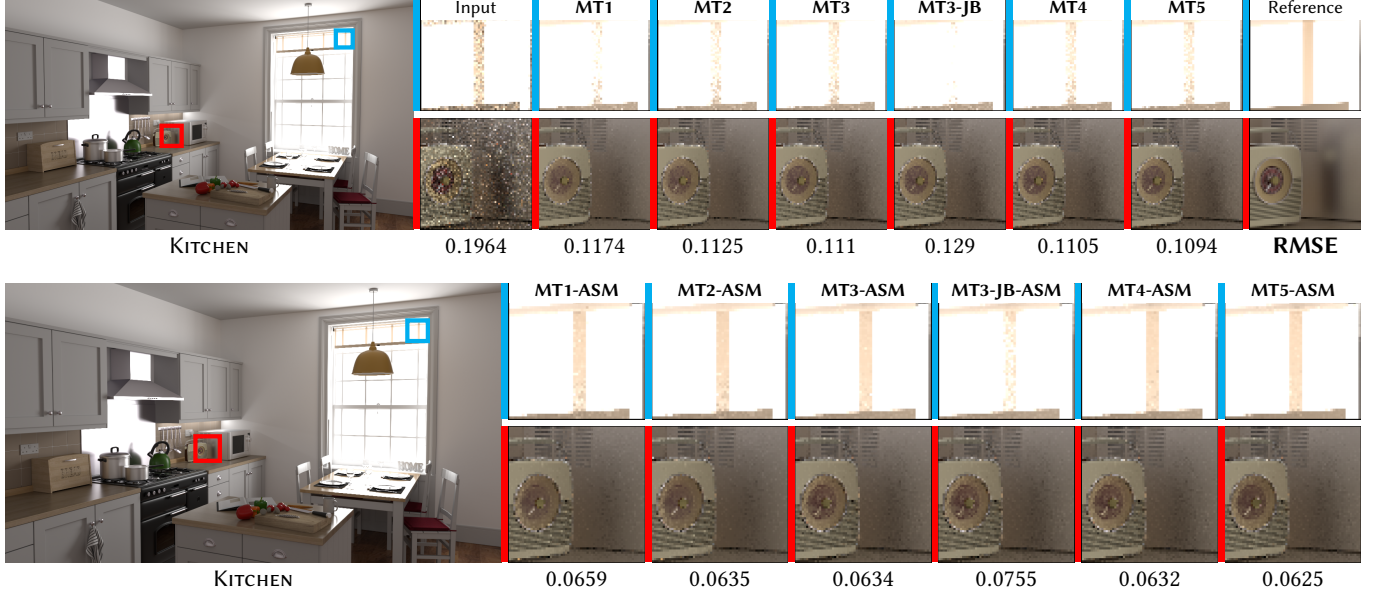


Fig. S16. Comparison of denoising quality using different multi-transform parameter sets, with (ASM) and without adaptive sampling, for KITCHEN rendered with 64 SPP (average for ASM). We have used our MCL denoiser and the following transformation sets—MT1: {0.5}, MT2: {0.5, 0.75}, **MT3** (default): {0.25, 1}, MT4: {0.25, 0.5, 0.75, 1}, MT5: {0.01, 0.25, 0.5, 0.75, 1, 3}, as well as MT3 using the Jarque–Bera statistic instead of the most-discriminative heuristic (MT3-JB).

Table S1. Denoising times (ms), including GPU up- and download, for the main results, obtained using an NVIDIA A40 GPU. Note that ASM+MCL timings include sample budget allocation every 32 SPP, and therefore increase with higher sample counts, whereas MCL denoising timings are independent of the input sample count. Our equal-time comparisons (Figs. S3, S17, and S18) indicate that the overhead is typically compensated by the resulting error reduction.

Fig.	Scene	SPP	Ours (MCL)	Ours (ASM+MCL)	Moon CI	OptiX	OIDN	ProDen	StatMC
1	VEACH EGG	2048	74.5	2518.8	36	116.3	23	4446.4	27.7
12	FUR BALL	64	77.3	157.1	37.4	114.3	22	4129.7	25.2
13	CAR	128	74.6	223.9	31.5	88.1	20.6	3678	25.4
14	BATHROOM	8192	77	9572.6	35.2	118.2	22.9	4504.4	26.8
S4	STRAIGHT HAIR	512	79.5	710.3	39.8	112.4	21.9	4086.9	27.4
S5	CURLY HAIR	128	79.5	248.4	40	114.6	22	4100	27.7
S6	LAMP	128	84.5	255.4	37.6	120.5	23.1	4452.6	30.8
S7	GLASS OF WATER	128	72.9	223.3	31.9	92.9	21.9	4008.9	26.3
S8	HOUSE	256	66.1	367.5	34.2	87.6	21.3	3628.2	19.1
S9	CLASSROOM	128	77.3	219.6	32.1	90.9	21.3	4020	25.7
S10	KITCHEN	128	69.1	214.6	38.9	88.5	20.6	3623	25.7
S11	ZERO-DAY	512	66.8	682.2	32.2	90.3	20.9	3916.1	26.7

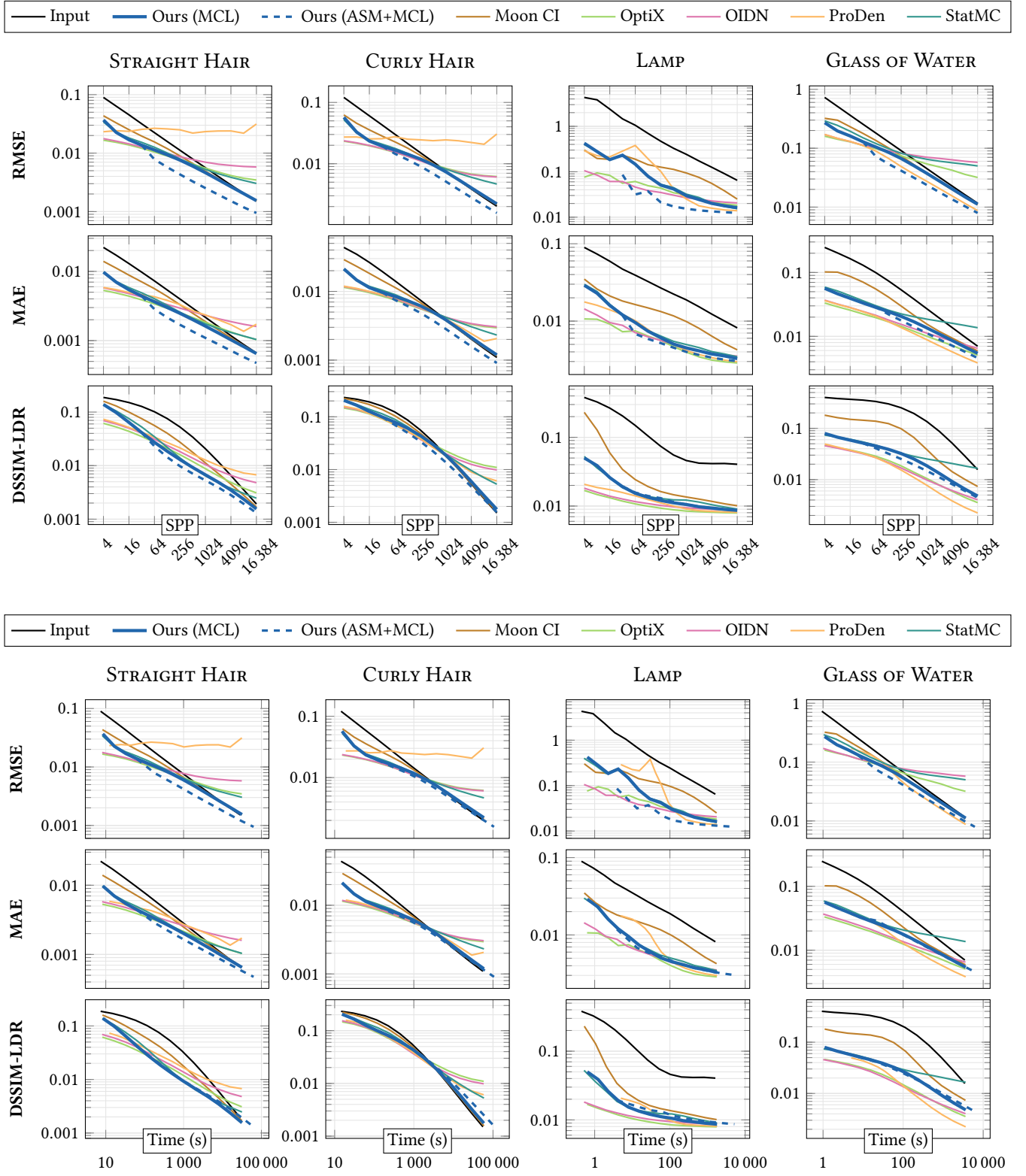


Fig. S17. Equal-sample (top) and equal-time (bottom) comparisons for the results presented in the supplementary document (part 1).

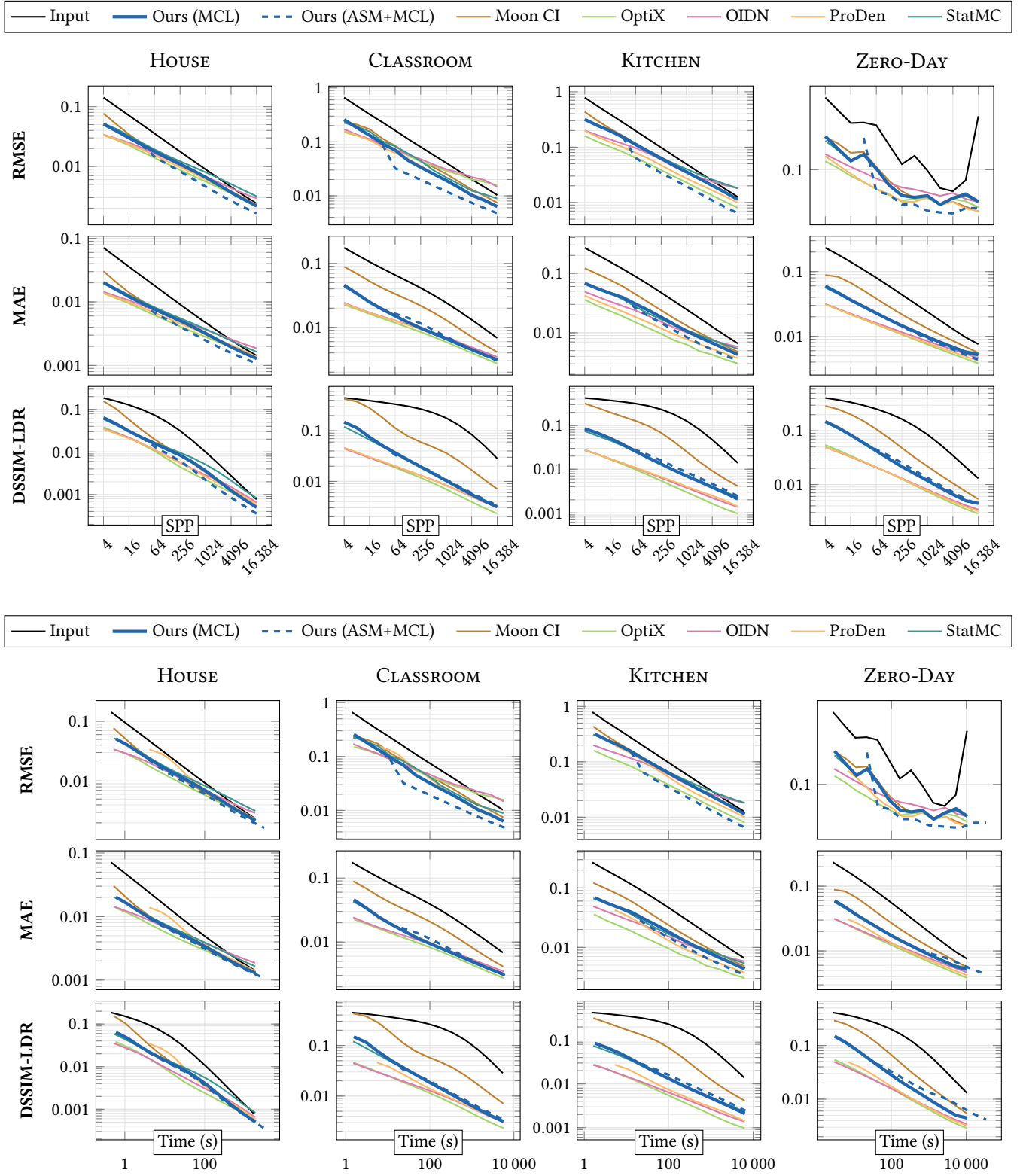


Fig. S18. Equal-sample (top) and equal-time (bottom) comparisons for the results presented in the supplementary document (part 2).