Learning sRGB-to-Raw-RGB De-rendering with Content-Aware Metadata

Seonghyeon Nam[1*]   Abhijith Punnappurath[2]   Marcus A. Brubaker[1,2]   Michael S. Brown[2]
[1]York University   [2]Samsung AI Center – Toronto
{shnnam,mab}@eecs.yorku.ca, {abhijith.p,michael.b1}@samsung.com

Abstract

Most camera images are rendered and saved in the standard RGB (sRGB) format by the camera's hardware. Due to the in-camera photo-finishing routines, nonlinear sRGB images are undesirable for computer vision tasks that assume a direct relationship between pixel values and scene radiance. For such applications, linear raw-RGB sensor images are preferred. Saving images in their raw-RGB format is still uncommon due to the large storage requirement and lack of support by many imaging applications. Several "raw reconstruction" methods have been proposed that utilize specialized metadata sampled from the raw-RGB image at capture time and embedded in the sRGB image. This metadata is used to parameterize a mapping function to de-render the sRGB image back to its original raw-RGB format when needed. Existing raw reconstruction methods rely on simple sampling strategies and global mapping to perform the de-rendering. This paper shows how to improve the de-rendering results by jointly learning sampling and reconstruction. Our experiments show that our learned sampling can adapt to the image content to produce better raw reconstructions than existing methods. We also describe an online fine-tuning strategy for the reconstruction network to improve results further.

1. Introduction

For many low-level computer vision tasks, it is desirable to have access to the camera's raw-RGB sensor image whose pixel values have a linear relationship with scene radiance [9, 29, 39]. In addition, photo-editing operations, such as white-balance adjustment or color manipulation, are more accurate when applied on raw-RGB images [19]. However, most images are still saved in the standard RGB (sRGB) format. sRGB images are raw-RGB images that have been rendered by the camera's image signal processor (ISP). The nonlinear photo-finishing routines applied by the ISP break the well-behaved relationship to scene radiance

---

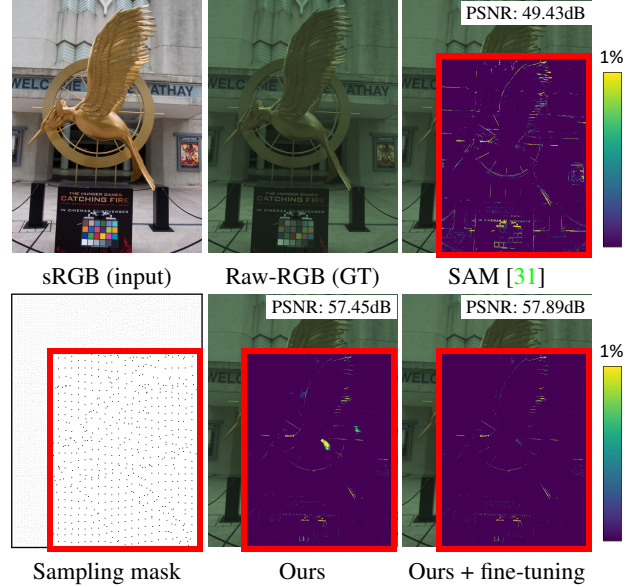*Work done while an intern at the Samsung AI Center – Toronto.

Figure 1. Overview of our paper. We address the problem of de-rendering an sRGB image to a raw-RGB image using a metadata saved along with the sRGB image. At capture time, we sample the raw-RGB values at the locations in a sampling mask, and save them as a metadata. When the raw-RGB image is needed, we reconstruct the full raw-RGB image from the sRGB image with the metadata. We propose an end-to-end deep learning framework to achieve it. With our approach, the reconstruction can be further improved by online fine-tuning.

present in the original raw-RGB image. A solution to recover the raw-RGB values is to "de-render" the sRGB image back to its raw-RGB format [4]. Among the various methods to de-render an image, the most accurate are those that collect samples from the original raw-RGB image at capture time and embed these samples inside the sRGB image as specialized metadata [28, 31]. These prior methods rely on uniform sampling of the raw-RGB image and simple mapping functions to de-render from sRGB to raw-RGB.

Contribution. We propose a deep-learning framework to address the sRGB de-rendering task using metadata sampled from the raw-RGB at capture time. In particular, we

demonstrate how sampling and reconstruction can both be learned in an end-to-end framework. Sampling is performed in a content-aware manner based on superpixel-based max-pooling and subsequently used by the reconstruction network. In addition, the reconstruction network incorporates an online fine-tuning approach to improve the performance at inference time. An example is shown in Fig. 1. We demonstrate the effectiveness of our method on the raw reconstruction task using 1.5% of raw-RGB pixels saved in the metadata and show we can achieve state-of-the-art performance. Additionally, we show the applicability of our method to other image recovery tasks by applying our sampling/reconstruction framework to bit-depth recovery.

2. Related Work

Algorithms intended to de-render sRGB images can be categorized into those that save specialized metadata along with the sRGB file at capture time and blind methods that require no additional information. We examine the metadata-based approaches in greater detail since these are more closely connected to our work. We also briefly survey algorithms for bit-depth recovery.

Blind raw reconstruction. Raw reconstruction is closely connected to the problem of radiometric calibration. Early digital cameras did not provide access to the sensor raw-RGB image. As a result, early radiometric calibration methods did not attempt to recover the raw-RGB values accurately. Instead, they focused on linearizing the sRGB data such that the digital values had a linear relationship to scene radiance. Radiometric calibration methods (e.g., [12, 15, 26]) employed simplistic models, such as a simple 1D response function per color channel.

As access to the raw sensor image became more commonplace, radiometric calibration was replaced by raw reconstruction, where the goal was to recover the original raw-RGB sensor data. The simple camera response function was replaced with more complex models [8, 9, 14, 19] to describe the various processing stages of the ISP. However, these methods are based on careful calibration procedures that must be repeated per camera and sometimes even per camera setting. Even recent deep learning methods (e.g., [25, 27]) are faced with similar issues in that large amounts of training data has to be captured for each camera, and the trained models are specific to that camera. Generic de-rendering methods, such as [4, 21], that assume a standard set of ISP operations are not very accurate because they cannot model the camera-specific operations.

Raw reconstruction with metadata. Another strategy for de-rendering is methods that save additional metadata in the sRGB to assist with the de-rendering process. For example, Yuan and Sun [39] propose storing a small raw image as additional metadata. The small raw image could be up-sampled to full resolution at edit time using the sRGB as a guide image. Work by Nguyen and Brown [28, 29] computed and stored metadata in the form of estimated parameters used to model the typical operations performed by the ISP. These estimated parameters added a 64 KB overhead and can reconstruct the raw image from the sRGB image. However, they assume that the mapping from sRGB to raw is global and ignore important ISP operations, such as local tone mapping. Punnappurath and Brown [31] recently proposed to use a small set of uniformly sampled raw values as metadata. They propose a spatially aware recovery algorithm that makes their method robust to local tone mapping and other non-global ISP manipulation. However, their raw reconstruction function is based on interpolation using a 5D radial basis function, which is slow in practice. Our method is closely related to the work in [31]; however, we do not restrict sampling to a uniform grid. Moreover, we learn both sampling and reconstruction in an end-to-end manner.

Bit-depth recovery. The bit-depth recovery problem shares similarities to the de-rendering problem and is discussed here. In particular, the camera has applied a nonlinear process (in this case, bit quantization), and the goal is to recover the original pixel values in their full precision. Traditional bit-depth recovery algorithms (e.g., [10, 17, 23, 35, 36] rely on heuristics for estimating missing bit precision based on the input image's structure. More recently, deep learning-based approaches learn how to recover missing bits [7, 16, 22, 24, 30, 33, 37, 40] based on training data before and after bit quantization. To the best of our knowledge, the use of metadata has not been explored for bit-depth recovery. As an extension of our work, we show that our proposed de-rendering framework can be adapted to bit-depth recovery with no changes in network architecture.

3. De-rendering Framework

We begin with a high-level description of our framework followed by details on the sampling and reconstruction components.

Let $\mathbf{x}$ and $\mathbf{y}$ denote an sRGB image and a raw-RGB image, respectively. Conventionally, the raw reconstruction problem has been mostly formulated by finding the mapping $\mathbf{y} = f(\mathbf{x})$ using only the sRGB image as input. For metadata approaches, the mapping $\mathbf{y} = f(\mathbf{x}; \mathbf{s}_y)$ is inferred by exploiting a small number of pixels $\mathbf{s}_y$ sampled from the raw-RGB image and saved in the metadata of the sRGB image. The pixels are usually sampled by a pre-defined method, such as uniform sampling applied to all images globally. Our goal is to learn both the sRGB-to-raw-RGB mapping and the sampling function, which is formally described as $\hat{\mathbf{y}} = f(\mathbf{x}; \hat{\mathbf{s}}_y = g(\mathbf{x}, \mathbf{y}))$, where $g(\mathbf{x}, \mathbf{y})$ is a learnable sampling function.

Fig. 2 shows an overview of our framework. We model two functions f and g as U-Net-based [32] deep neural networks, and train them in an end-to-end manner. At training
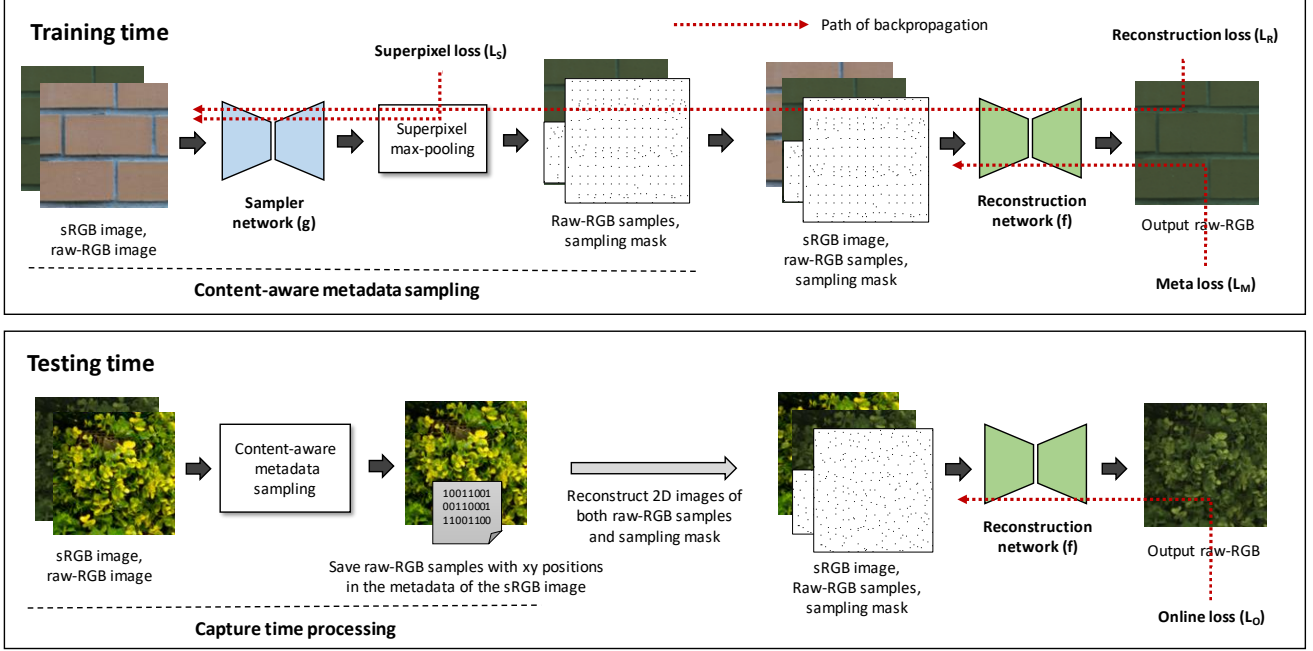
Figure 2. Overview of our sRGB-to-raw de-rendering framework. At training time, we train a sampler network g and reconstruction network f in an end-to-end manner. g predicts a binary sampling mask used to sample raw-RGB values, while f recovers the full raw-RGB image from a full sRGB image with the sampled raw-RGB values. Particularly, our sampling mask is generated by a superpixel-based max-pooling. At testing time, g is used to save content-aware metadata along with an sRGB image. When needed, the full raw-RGB image is reconstructed by f. We fine-tune f using the sparse raw-RGB samples in the metadata on the fly to further improve the performance.
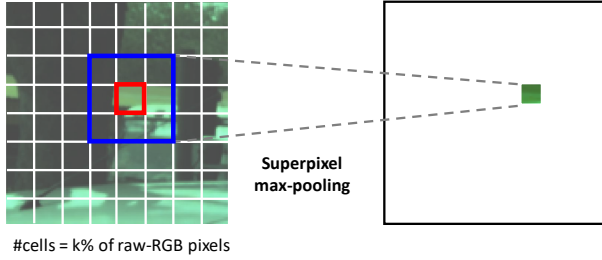

Figure 3. Illustration of the superpixel max-pooling. For all pixels in the blue box, their association to the red cell is learned by the superpixel loss. The superpixel max-pooling is to sample the highest association score among the pixels in the blue box. Even though we sample a pixel from a regular box, the sampling heavily relies on the association score learned by the superpixel loss.

time, we first sample $k\%$ of pixels from a raw-RGB image. Specifically, the sampler network g takes both a raw-RGB and an sRGB image as input and predicts a binary sample map $\mathbf{s}$ in which sampled pixels are assigned to 1. To effectively compute samples, the sampler network also learns to divide the raw-RGB image into superpixels and selects samples by per-superpixel max-pooling. In the reconstruction network, both the sRGB image and sampled raw-RGB pixels with their corresponding mask are fed into the net-

work to recover the full raw-RGB image. The two networks are jointly trained by minimizing the pixel-wise distance between the output raw-RGB image and ground truth.

The inference time scenario is composed of two stages. At capture time, we use the sampler network g to sample the raw-RGB image. These samples are stored as metadata in the sRGB image as comments. To save memory, we only save the sampled RGB values with their pixel positions in the metadata. When needed, the raw-RGB image is reconstructed by the reconstruction network f with the saved metadata in the sRGB image. Even though the pre-trained reconstruction network produces high-quality raw-RGB images, it can be fine-tuned by the sparse raw-RGB samples to further improve the performance on the test data.

3.1. Content-Aware Metadata Sampling

The goal of our content-aware metadata sampling is to find the optimal raw-RGB samples based on the content of the image. To this end, our key idea is to divide a raw-RGB image into superpixels and select the best pixel in each superpixel as a sample for metadata. We found that it is beneficial for the reconstruction network to choose the raw-RGB samples that are well distributed over the space of a raw-RGB image. As a result, we split the xy-RGB space of raw-RGB images into multiple subspaces using a superpixel

segmentation and collect the representative pixels.

Specifically, inspired by [38], our sampler network first computes superpixels directly from the input. As shown in Fig. 3, we divide a raw-RGB image into uniform grid cells. The network predicts the association scores $q_c(\mathbf{p})$ for each pixel $\mathbf{p}$ that indicate how likely a pixel $\mathbf{p}$ belongs to a grid cell c. For computational efficiency, only the nine neighboring cells in the blue box are considered for computing the association to the red-highlighted cell. The association map is learned by optimizing the following superpixel segmentation loss:

$$
\begin{aligned}
L_S = \; & \alpha \sum_{\mathbf{p}} \|\mathbf{x}(\mathbf{p}) - \hat{\mathbf{x}}(\mathbf{p})\|_2^2 \\
& + (1 - \alpha) \sum_{\mathbf{p}} \|\mathbf{y}(\mathbf{p}) - \hat{\mathbf{y}}(\mathbf{p})\|_2^2 \\
& + \frac{m^2}{S^2} \sum_{\mathbf{p}} \|\mathbf{p} - \hat{\mathbf{p}}\|_2^2,
\end{aligned}
\tag{1}
$$

where $\hat{\mathbf{x}}(\mathbf{p})$ is a reconstructed RGB value of $\mathbf{x}(\mathbf{p})$ computed by the following equations:

$$
\mathbf{u}_c = \frac{\sum_{\mathbf{p} \in \mathcal{N}_c} \mathbf{x}(\mathbf{p}) \cdot q_c(\mathbf{p})}{\sum_{\mathbf{p} \in \mathcal{N}_c} q_c(\mathbf{p})}, \; \hat{\mathbf{x}}(\mathbf{p}) = \sum_{c} \mathbf{u}_c \cdot q_c(\mathbf{p}). \tag{2}
$$

In the equation, $\mathbf{u}_c$ is the feature vector of the center of the superpixel c and $\mathcal{N}_c$ is a set of all pixels in the nine surrounding cells of a cell c. Similar to SLIC [2], the loss in Eq. (1) enforces that the pixels in each superpixel are not deviated much from the center $\mathbf{u}_c$. $\hat{\mathbf{y}}(\mathbf{p})$ and $\hat{\mathbf{p}}$ are computed by the same equations. In Eq. (1), m and S are the weight parameters in [38]. Unlike the loss in [38] that uses semantic segmentation labels, our loss optimizes the RGB color distance in both the sRGB and raw-RGB images. Our loss forces the network to consider how to sample the sRGB and raw-RGB images jointly. We add a hyperparameter α to balance the sRGB and raw-RGB terms.

To select $k\%$ of samples, we set the number of uniform grid cells to $k\%$ of the number of raw-RGB pixels. We then choose the representative pixel for each grid cell that provides the maximum q in the pixels of the nine neighborhood cells. The per-cell max-pooling is formulated as

$$
\mathbf{p}_c^* = \arg\max_{\mathbf{p} \in \mathcal{N}_c} q_c(\mathbf{p}). \tag{3}
$$

We compute a binary sampling mask $\mathbf{m}$ to feed the samples to the reconstruction network in the form of a 2D image, which is formally described as

$$
\mathbf{m}(\mathbf{p}) = \begin{cases} 1, & \text{if } \mathbf{p} \in \{\mathbf{p}_0^*, \mathbf{p}_1^*, ..., \mathbf{p}_c^*\} \\ 0, & \text{otherwise.} \end{cases} \tag{4}
$$

The sample map $\mathbf{s}_y$ is simply computed by multiplying $\mathbf{m}$ and $\mathbf{y}$, which is described as $\mathbf{s}_y = \mathbf{m} \otimes \mathbf{y}$ where $\otimes$ is a
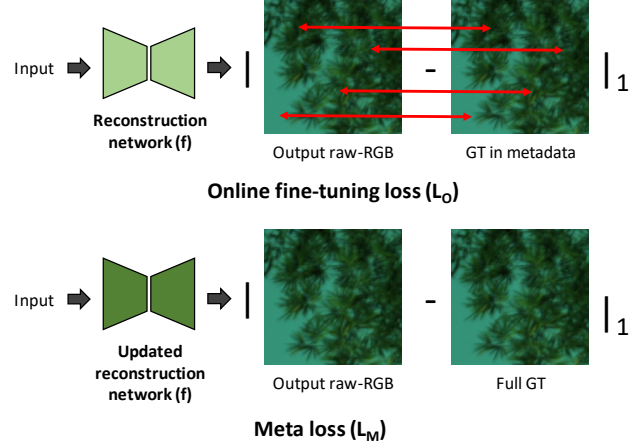

Figure 4. Illustration of the online fine-tuning loss L_O and meta loss L_M. The L_O is computed only at the positions that have ground-truth raw-RGB samples in metadata. To compute L_M, f is first updated by L_O, and then the loss for the full output image is computed.

Hadamard product. The gradient is back-propagated only through $q(\mathbf{p})$ such that $\mathbf{m}(\mathbf{p}) = 1$ using a straight-through estimator [3]. Note that our method does not guarantee that the number of samples equals the $k\%$ of the number of pixels because multiple cells may choose the same pixel after max-pooling.

3.2. sRGB-to-Raw-RGB De-rendering Using Metadata

The sRGB-to-raw-RGB de-rendering task is formulated as an image-to-image transform learned by the reconstruction network f. To exploit the sparse raw-RGB samples in the metadata for reconstruction, we concatenate $\mathbf{x}$, $\mathbf{s}_y$, and $\mathbf{m}$ followed by feeding them to f. With these inputs, the network can infer the sRGB-to-raw-RGB mapping for all pixels based on the sparse sRGB-raw-RGB pairs and the full sRGB image. Both the sampler and reconstruction networks are jointly trained by the pixel distance loss described as

$$
L_R = \sum_{\mathbf{p}} \|\hat{\mathbf{y}}(\mathbf{p}) - \mathbf{y}(\mathbf{p})\|_1, \tag{5}
$$

where $\hat{\mathbf{y}} = f(\mathbf{x}, \mathbf{s}_y, \mathbf{m})$.

3.3. Online Fine-Tuning at Inference Time

Another benefit of storing raw-RGB samples in the metadata is that they can be used to fine-tune the pre-trained reconstruction network on the fly for a test image to improve the performance further. As shown in Fig. 2 and Fig. 4, we minimize the pixel-wise distance loss only for the pixels that have their corresponding ground truth at inference

time, which is formally described as

$$L_O = \sum_{\mathbf{p}} \mathbf{m}(\mathbf{p}) \cdot \|\hat{\mathbf{y}}(\mathbf{p}) - \mathbf{y}(\mathbf{p})\|_1. \qquad (6)$$

Meta-learning for optimizing fine-tuning. Our reconstruction network can be further optimized at training time to be generalized to fine-tuning on test-time examples. Intuitively, we expect that the overall error is minimized after the network is fine-tuned to fit the sparse samples. We can encourage our network to be receptive to fine-tuning by adding another loss term at training time. Specifically, we first compute the reconstruction output using the metadata samples $\tilde{\mathbf{s}}$ and $\tilde{\mathbf{m}}$, which is described as $\tilde{\mathbf{y}}_\theta = f_\theta(\mathbf{x}, \tilde{\mathbf{s}}_y, \tilde{\mathbf{m}})$, where θ is the parameters of f. Then, we update L_O in Eq. (6) by several gradient descent steps using the following update rule: $\theta' = \theta - \beta \nabla_\theta L_O(\tilde{\mathbf{m}}, \tilde{\mathbf{y}}_\theta)$, where β is a learning rate. We finally compute the pixel distance loss between the output of the updated model and ground truth for all pixels as shown in Fig. 4, which is formally described as

$$L_M = \sum_{\mathbf{p}} \|\tilde{\mathbf{y}}_{\theta'}(\mathbf{p}) - \mathbf{y}(\mathbf{p})\|_1. \qquad (7)$$

We have found that feeding the learned samples from the sampler network to this loss degrades performance. The reason is that the goal of two losses conflicts: the main loss L_R is optimized to overfit training batches while L_M seeks generalization. Therefore, we use different data to compute the loss. Specifically, we use random samples for $\tilde{\mathbf{s}}$ and $\tilde{\mathbf{m}}$. This strategy alleviates the performance degradation when optimizing both losses and forces the network to cope with a variety of sampling maps, which is also helpful for generalization. Our formulation shares a similar spirit with MAML [13], a meta-learning approach, in that both are to find generalizable parameters that result in better performance at fine-tuning. Our goal is to improve the overall image quality even after overfitting the network to the $k\%$ of pixels. We use FOMAML [13], a first-order approximation of gradients, to compute the gradients of the meta loss.

3.4. Training Objective

The final training objective at training time is composed of the reconstruction loss, superpixel loss, and meta loss, which is described as

$$L_{Total} = L_R + \lambda_S L_S + \lambda_M L_M, \qquad (8)$$

where λ_S and λ_M are hyperparameters. At testing time, the online fine-tuning is performed by optimizing the online optimization loss in Eq. (6).

4. Experiments

4.1. Experimental Settings

Dataset. To test the effectiveness of our method, we use the NUS dataset [11], which contains raw images from several different cameras. For our experiments, we use three cameras—Samsung NX2000, Olympus E-PL6, and Sony SLT-A57—containing 202, 208, and 268 raw images, respectively. We demosaic the raw Bayer image using standard bi-linear interpolation to obtain a 3-channel raw-RGB image. We then process the raw-RGB image using a software ISP emulator [18] to render the corresponding sRGB image. This rendering mimics the photo-finishing applied by the camera. We randomly split images from each camera into training, validation, and test sets. In addition, we crop all images into overlapping 128×128 patches.

In addition to the raw images, the NUS dataset also contains sRGB-JPEG images rendered by each individual camera's ISP. We also performed experiments where we used these sRGB images instead of the software ISP emulator [18]. These results are reported in Section A.

Baselines. We compare our method with two metadata-based raw reconstruction methods: RIR [28] and SAM [31]. The RIR method stores the parameters of global operations of an ISP as metadata. The SAM method, which is most similar to our approach, saves uniformly sampled raw-RGB values along with an sRGB image. Since the source codes of the methods are not publicly available, we implement them to reproduce results. For SAM, we use the same sampling ratio used in our method.

Implementation details. As a backbone of both the sampler and reconstruction networks, we use a U-Net [32] architecture. We train our networks using the Adam optimizer [20] with a learning rate of 0.001 and a batch size of 128 for 120 epochs. In the superpixel loss, we use 0.2 and 10 for α and m, respectively. In the meta loss, we use five gradient descent steps for the inner update with a learning rate of 0.001. We also set λ_S and λ_M to 0.0001 and 0.01 for all cameras except for the Sony; we use λ_M of 0.001 for the Sony set. At testing time, we fine-tune the network for ten iterations with a learning rate of 0.0001. In all experiments, we sample 1.5% of pixels from a raw-RGB image, which are 256 pixels in a 128×128 patch. We train a model per camera since raw images are in a sensor-dependent color space. Our code and pre-trained models are available at https://github.com/SamsungLabs/content-aware-metadata.

4.2. Experimental Results

Table 1 shows a quantitative comparison of the three cameras in the NUS dataset. For a fair comparison, we evaluate the performance of our method on the fully reconstructed raw-RGB images. As can be seen, our method out-

Method	Fine-tuning	Samsung NX2000		Olympus E-PL6		Sony SLT-A57	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
RIR [28]	N/A	45.66	0.9939	48.42	0.9924	51.26	0.9982
SAM [31]	N/A	47.03	0.9962	49.35	0.9978	50.44	0.9982
Ours	No	48.08	0.9968	50.71	0.9975	50.49	0.9973
Ours	Yes	**49.57**	**0.9975**	**51.54**	**0.9980**	**53.11**	**0.9985**

Table 1. Quantitative evaluation on raw reconstruction.

Figure 5. Qualitative comparison. Each two rows show results on Samsung NX2000, Olympus E-PL6, and Sony SLT-A57, respectively.

performs the baseline methods by a large margin after fine-tuning. The PSNRs of our method without fine-tuning are also higher than those of the baselines on the Samsung and Olympus data. The RIR achieved high performance on the Sony data, while the method is the worst on the Samsung and Olympus data. We speculate that the Sony camera's raw-RGB-to-sRGB mapping has little effect on local processing. Thus, the de-rendering is modeled well by a global approach. Nevertheless, our method achieves the best after fine-tuning. Even though the fine-tuning heavily relies

on a very small subset of raw-RGB pixels, the training is generally effective. We attribute this to the inductive bias of self-similarity in CNNs [34]. As convolution filters are shared in all spatial locations, the training signals at sparse locations can be propagated to neighborhood pixels.

Fig. 5 shows a qualitative comparison. From top to bottom, we show two results each on Samsung NX2000, Olympus E-PL6, and Sony SLT-A57, respectively. The baseline results have high errors, mostly on edges, as their models are not sophisticated. The RIR relies on global operators of

Method	Fine-tuning	PSNR	SSIM
No metadata	N/A	47.67	0.9913
Uniform		49.58	0.9940
Random	No	49.68	0.9940
Ours		**50.64**	**0.9942**
Uniform		52.59	0.9960
Random	Yes	52.55	0.9957
Ours		**53.32**	**0.9961**

Table 2. Comparison of different sampling methods. All methods share the same reconstruction network.
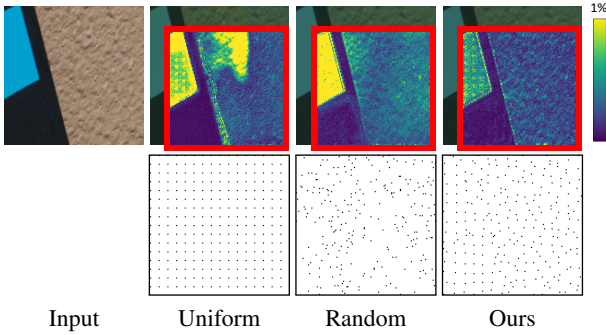

Input   Uniform   Random   Ours

Figure 6. Comparison of different samplings. The top and bottom rows show the error maps and sampling maps, respectively.

Method	PSNR	SSIM
sRGB	49.10	0.9927
RAW	49.72	0.9933
sRGB + RAW	**50.15**	**0.9943**
Free-form max-pooling	47.75	0.9911
Superpixel max-pooling	**50.15**	**0.9943**
W/o meta loss	50.15	0.9943
W/o meta loss + fine-tuning	53.07	0.9959
W/ meta loss	50.64	0.9942
W/ meta loss + fine-tuning	**53.32**	**0.9961**

Table 3. Ablation study on our method. We compare different inputs fed into the sampling process and different pooling approaches without using a meta loss. We also analyze the effectiveness of the meta loss.
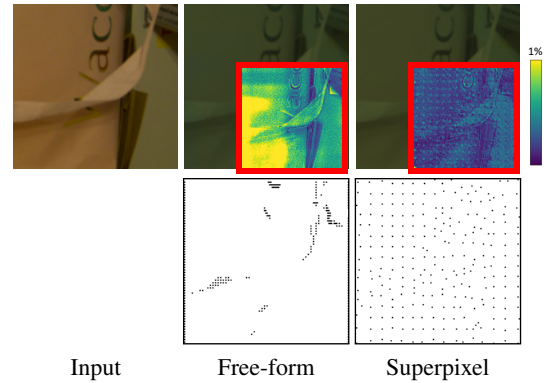

Input   Free-form   Superpixel

Figure 7. Ablation study on the superpixel max-pooling. The two rows show error maps and sampling masks, respectively.

an ISP, but the edges are usually processed further by local tone mappings. Even though the SAM can model spatially varying color mappings, its uniform sampling is not enough to store complex information around edges. In contrast, our method adaptively samples raw-RGB values according to the scene structure, improving the overall performance.

4.3. Discussion

Uniform vs. random vs. content-aware sampling. In Table 2, we compare different sampling approaches, including uniform and random sampling. We also compare against a baseline where there is no sampling i.e., the reconstruction network is provided no additional metadata. We average the PSNR and SSIM results of patches from the three cameras. Uniform sampling chooses samples from a 2D grid of the image, while random sampling samples $k\%$ of pixels randomly in an image. Both the samplings are independent of the image content. As shown in the table, exploiting metadata significantly improves the quality of raw reconstruction regardless of sampling methods, which demonstrates that it is beneficial to save a small number of raw-RGB pixel values in the metadata of an sRGB image. The uniform and random samplings are simple but effective approaches as they choose samples evenly in the entire space of pixels in an image. As shown in Fig. 6, however,

the performance is limited since they are unable to select the samples particularly useful for reconstruction. On the other hand, our content-aware sampling outperforms the simple approaches by a large margin because our method not only samples pixels evenly but also considers their effectiveness on reconstruction. An ablation on the sampling rate k is provided in Section B.

Ablation study. To demonstrate individual components in our method, we conduct an ablation study, as shown in Table 3. We first try different inputs to the sampler network and superpixel loss: sRGB image, raw-RGB image, and both images. As can be seen, it is beneficial to use raw-RGB images as input and then use sRGB images to find samples from the raw-RGB images. However, the network achieves the highest scores when using both images, indicating that the sRGB images still provide useful information for better sampling when both images are jointly used.

We also compare our superpixel-based sampling with a naive sampling approach. For the free-form max-pooling in the table, we train a sampler network with a single-

Method	PSNR	SSIM
CA [35]	34.74	0.9317
ACDC [36]	34.68	0.9152
IPAD [23]	34.91	0.9345
BitNet [7]	38.48	0.9657
BE-CALF [24]	38.94	0.9680
Ours	39.57	0.9719
Ours + fine-tuning	**39.73**	**0.9721**

Table 4. Quantitative comparison of bit-depth recovery (4-to-8-bit) methods on the Kodak dataset [1].

channel sigmoid output and extract the top $k\%$ of pixels that have large sigmoid values. As shown in the table, the performance of the free-form sampling significantly degrades compared with the superpixel-based sampling. Fig. 7 shows a qualitative comparison. Since the free-form sampling is unconstrained, most samples are clustered in the same region. In contrast, our superpixel-based sampling enables the network to sample pixels at various locations while covering the full spatial range of the image.

Lastly, we conduct an ablation study on the meta loss. As shown in Table 3, the meta loss improves the performance after fine-tuning and the direct result from the network, which demonstrates that the loss forces the reconstruction network to learn its weights generalizable to unseen test cases. With the loss, the network can improve its performance over 3dB compared to the direct network output without the loss using sparse raw-RGB samples.

Limitations. Unlike uniform and random sampling, our method is required to run a deep neural network to sample pixels at capture time, which is an additional computational cost on the device. We have not tested edge cases, such as high compression, very noisy low-light images, or under/overexposure issues in raw. Since convolutional neural networks generally fit well with natural images that are spatially smooth, it is still unclear how well our reconstruction network can process sparse sampling masks. Investigating efficient deep architectures for sparse samples is an interesting direction of research in the future.

4.4. Other application: Bit-Depth Recovery

Our framework for sampling and reconstruction can also be applied to the problem of bit-depth recovery. To test on this task, we use two publicly available datasets for training: MIT-Adobe 5K [6] and Sintel [5]. We synthesize pairs of 4-bit and 8-bit images to train a 4-to-8-bit recovery task. We use the 1.5% of pixels of an 8-bit image as metadata for our method and run the update steps for fine-tuning.

Table 4 and Fig. 8 show a quantitative and qualitative comparison with blind bit-depth recovery approaches. We
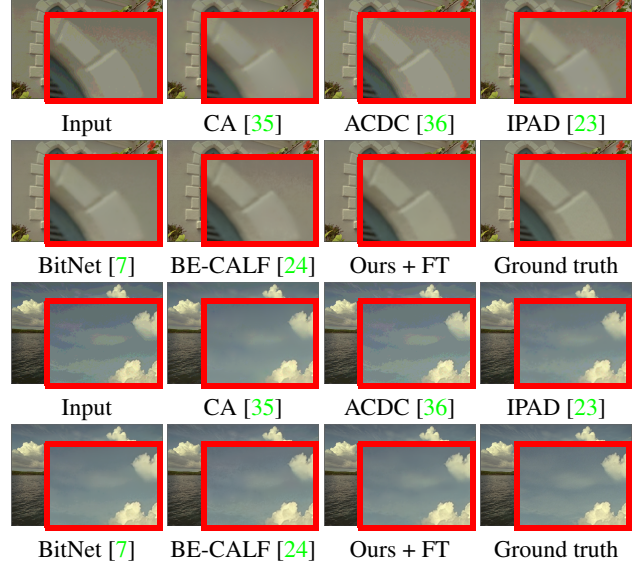

Figure 8. Qualitative comparison of bit-depth recovery algorithms on the Kodak dataset [1]. Zoom in for better visibility.

evaluate all methods on the Kodak dataset [1]. As expected, our method outperforms the baselines using a small number of pixels in the metadata. Note that our method is not explicitly designed for bit-depth recovery. Our reconstruction network is a generic U-Net, while BitNet [7] and BE-CALF [24] use networks particularly designed for the task. This experiment demonstrates the applicability of our metadata framework to other image processing tasks.

5. Conclusion

We have presented a method for sRGB image de-rendering that recovers the original raw-RGB images with the assistance of a small amount of metadata that is sampled from the raw-RGB image at capture time. Our approach learns both the sampling and reconstruction network in an end-to-end manner. Moreover, we train the reconstruction network to lend itself for further fine-tuning from the sparse metadata samples. We show significant improvements over the existing state-of-the-art approaches that also use metadata. Finally, we use our framework for the related task of bit-depth recovery and show compelling results.

Acknowledgments

References

[1] Kodak lossless true color image suite. http://r0k.us/graphics/kodak, 1999. 8

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE TPAMI*, 34(11):2274–2282, 2012. 4, 11

[3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv*, 2013. 4

[4] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T. Barron. Unprocessing images for learned raw denoising. In *CVPR*, 2019. 1, 2

[5] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012. 8

[6] Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. Learning photographic global tonal adjustment with a database of input/output image pairs. In *CVPR*, 2011. 8

[7] Junyoung Byun, Kyujin Shim, and Changick Kim. BitNet: Learning-based bit-depth expansion. In *ACCV*, 2018. 2, 8

[8] Ayan Chakrabarti, Daniel Scharstein, and Todd E. Zickler. An empirical camera model for internet color vision. In *BMVC*, 2009. 2

[9] Ayan Chakrabarti, Ying Xiong, Baochen Sun, Trevor Darrell, Daniel Scharstein, Todd Zickler, and Kate Saenko. Modeling radiometric uncertainty for vision with tone-mapped color images. *IEEE TPAMI*, 36(11):2185–2198, 2014. 1, 2

[10] Cheuk Cheng, Oscar Au, Chun Hung Liu, and Ka Yip. Bit-depth expansion by contour region reconstruction. In *IEEE International Symposium on Circuits and Systems*, 2009. 2

[11] Dongliang Cheng, Dilip K Prasad, and Michael S Brown. Illuminant estimation for color constancy: Why spatial-domain methods work and the role of the color distribution. *JOSA A*, 31(5):1049–1058, 2014. 5, 11

[12] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *ACM SIGGRAPH*, 2008. 2

[13] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 5

[14] Han Gong, Graham D Finlayson, Maryam M Darrodi, and Robert B Fisher. Rank-based radiometric calibration. In *Color Imaging Conference*, 2018. 2

[15] Michael D. Grossberg and Shree K. Nayar. Determining the camera response from images: What is knowable? *IEEE TPAMI*, 25(11):1455–1467, 2003. 2

[16] Xianxu Hou and Guoping Qiu. Image companding and inverse halftoning using deep convolutional neural networks. *arXiv*, 2017. 2

[17] Masayuki Ikebe and Akira Mizuno. Bit-depth expansion for noisy contour reduction in natural images. In *ICASSP*, 2016. 2

[18] Hakki Can Karaimer and Michael S Brown. A software platform for manipulating the camera imaging pipeline. In *ECCV*, 2016. 5, 11

[19] Seon Joo Kim, Hai Ting Lin, Zheng Lu, Sabine Süsstrunk, Stephen Lin, and Michael S. Brown. A new in-camera imaging model for color computer vision and its application. *IEEE TPAMI*, 34(12):2289–2302, 2012. 1, 2

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv*, 2014. 5

[21] Samu Koskinen, Dan Yang, and Joni-Kristian Kämäräinen. Reverse imaging pipeline for raw RGB image augmentation. In *ICIP*, 2019. 2

[22] Jing Liu, Wanning Sun, and Yutao Liu. Bit-depth enhancement via convolutional neural network. In *Digital TV and Wireless Multimedia Communication*, 2018. 2

[23] Jing Liu, Guangtao Zhai, Anan Liu, Xiaokang Yang, Xibin Zhao, and Chang Wen Chen. IPAD: Intensity potential for adaptive de-quantization. *IEEE TIP*, 27(10):4860–4872, 2018. 2, 8

[24] Wanning Liu, Jing andß Sun, Yuting Su, Peiguang Jing, and Xiaokang Yang. BE-CALF: Bit-depth enhancement by concatenating all level features of dnn. *IEEE TIP*, 28(10):4926–4940, 2019. 2, 8

[25] Yu-Lun Liu, Wei-Sheng Lai, Yu Sheng Chen, Yi-Lung Kao, Ming-Hsuan Yang, Yung-Yu Chuang, and Jia-Bin Huang. Single-image HDR reconstruction by learning to reverse the camera pipeline. In *CVPR*, 2020. 2

[26] Tomoo Mitsunaga and Shree K. Nayar. Radiometric self calibration. In *CVPR*, 1999. 2

[27] Seonghyeon Nam and Seon Joo Kim. Modelling the scene dependent imaging in cameras with a deep neural network. In *ICCV*, 2017. 2

[28] Rang M. H. Nguyen and Michael S. Brown. RAW image reconstruction using a self-contained sRGB-JPEG image with only 64 KB overhead. In *CVPR*, 2016. 1, 2, 5, 6, 11, 13, 14, 15

[29] Rang M. H. Nguyen and Michael S. Brown. RAW image reconstruction using a self-contained sRGB–JPEG image with small memory overhead. *IJCV*, 126(6):637–650, 2018. 1, 2

[30] Abhijith Punnappurath and Michael S. Brown. A little bit more: Bitplane-wise bit-depth recovery. *TPAMI*, 2021. 2

[31] Abhijith Punnappurath and Michael S. Brown. Spatially aware metadata for raw reconstruction. In *WACV*, 2021. 1, 2, 5, 6, 11, 13, 14, 15

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 2, 5

[33] Yuting Su, Wanning Sun, Jing Liu, Guangtao Zhai, and Peiguang Jing. Photo-realistic image bit-depth enhancement via residual transposed convolutional neural network. *Neurocomputing*, 2019. 2

[34] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *CVPR*, pages 9446–9454, 2018. 6

[35] Pengfei Wan, Oscar C Au, Ketan Tang, Yuanfang Guo, and Lu Fang. From 2d extrapolation to 1d interpolation: Content adaptive image bit-depth expansion. In *ICME*, 2012. 2, 8

[36] Pengfei Wan, Gene Cheung, Dinei Florencio, Cha Zhang, and Oscar C Au. Image bit-depth enhancement via maximum a posteriori estimation of ac signal. *IEEE TIP*, 25(6):2896–2909, 2016. 2, 8

[37] Yi Xiao, Chao Pan, Yan Zheng, Xianyi Zhu, Zheng Qin, and Jin Yuan. Gradient-guided DCNN for inverse halftoning and image expanding. In *ACCV*, 2019. 2

[38] Fengting Yang, Qian Sun, Hailin Jin, and Zihan Zhou. Superpixel segmentation with fully convolutional networks. In *CVPR*, 2020. 4

[39] Lu Yuan and Jian Sun. High quality image reconstruction from RAW and JPEG image pair. In *ICCV*, 2011. 1, 2

[40] Yang Zhao, Ronggang Wang, Wei Jia, Wangmeng Zuo, Xiaoping Liu, and Wen Gao. Deep reconstruction of least significant bits for bit-depth expansion. *IEEE TIP*, 28(6):2847–2859, 2019. 2

Method	Fine-tuning	Samsung NX2000		Olympus E-PL6		Sony SLT-A57	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
RIR [28]	N/A	37.62	0.9696	42.19	0.9865	45.22	0.9916
SAM [31]	N/A	38.80	0.9725	43.15	0.9881	46.02	0.9921
Ours	No	40.80	0.9812	46.89	0.9938	48.51	0.9947
Ours	Yes	**41.59**	**0.9818**	**47.76**	**0.9944**	**49.58**	**0.9954**

Table A1. Quantitative evaluation on raw reconstruction using the camera ISP sRGB images.

Percentage Samples k	0%	0.4%	1.5%	6.25%
Samsung NX2000	38.86	48.56	49.57	50.31
Olympus E-PL6	42.30	50.62	51.54	52.20
Sony SLT-A57	44.79	51.09	53.11	53.44

Table A2. An ablation on the sampling rate k.

Appendix A. Results on camera ISP images

In Section 4, we had evaluated our raw reconstruction accuracy using the NUS [11] dataset with the sRGB images rendered using a software ISP emulator [18]. The NUS dataset also contains the sRGB-JPEG images rendered by each individual camera's hardware ISP. We used these sRGB images instead of the software ISP emulator [18], and the results are presented in Table A1. Our method generalizes well to different ISPs, and outperforms competitors. We do note that compared to Table 1, there is a drop in performance for all methods due to the more complex ISPs.

Appendix B. Other sampling rates

We report PSNR (dB) for our method (with fine-tuning) at different sampling rates in Table A2. The results for $k = 1.5\%$ are reproduced from Table 1. There is a significant improvement from 0% i.e., no metadata, to 0.4%. Performance improves with higher k values but at the expense of larger metadata size.

Appendix C. Comparison with SLIC

We performed an experiment where we replaced our learned superpixel with SLIC [2] for sampling, and trained our reconstruction network under the same settings. On the Samsung camera, we obtained PSNR/SSIM values of 45.94 / 0.9958 as against 49.57 / 0.9975 produced by our method, demonstrating the superiority of an end-to-end learnable superpixel sampler.

Appendix D. Additional experiments

In Fig. A1, we compare the error maps of our outputs before and after fine-tuning. Fig. A2, Fig. A3, and Fig. A4 show additional qualitative results on three cameras. For visibility, we omit output raw-RGB images and ground truth. Fig. A5 to Fig. A7 show visualizations of learned superpixels and sampling masks.
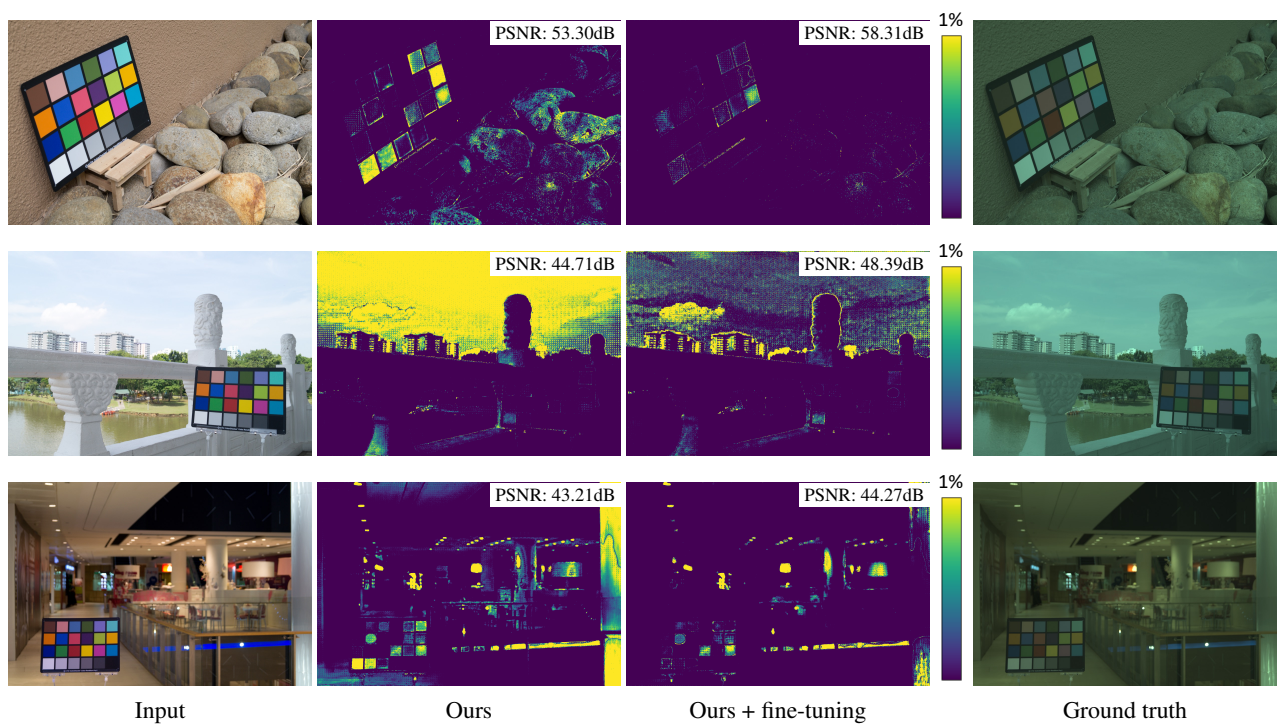
Input	Ours	Ours + fine-tuning	Ground truth

PSNR: 53.30dB · PSNR: 58.31dB

PSNR: 44.71dB · PSNR: 48.39dB

PSNR: 43.21dB · PSNR: 44.27dB

Figure A1. Qualitative evaluation of our online fine-tuning.

| Input | RIR [28] | SAM [31] | Ours + fine-tuning |

Figure A2. Qualitative comparison on Samsung NX2000.

| Input | RIR [28] | SAM [31] | Ours + fine-tuning |

Figure A3. Qualitative comparison on Olympus E-PL6.

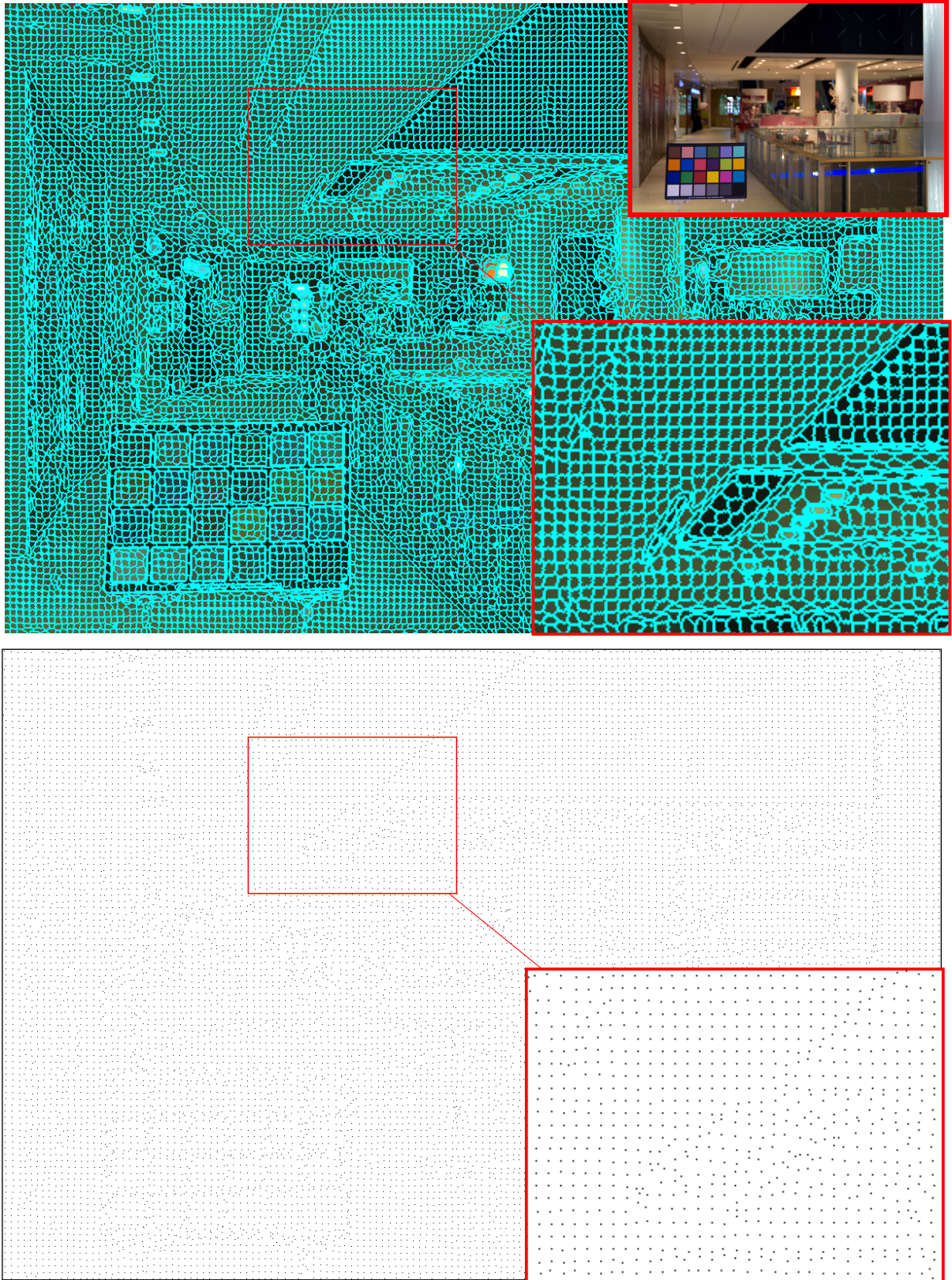| Input | RIR [28] | SAM [31] | Ours + fine-tuning |

Figure A4. Qualitative comparison on Sony SLT-A57.

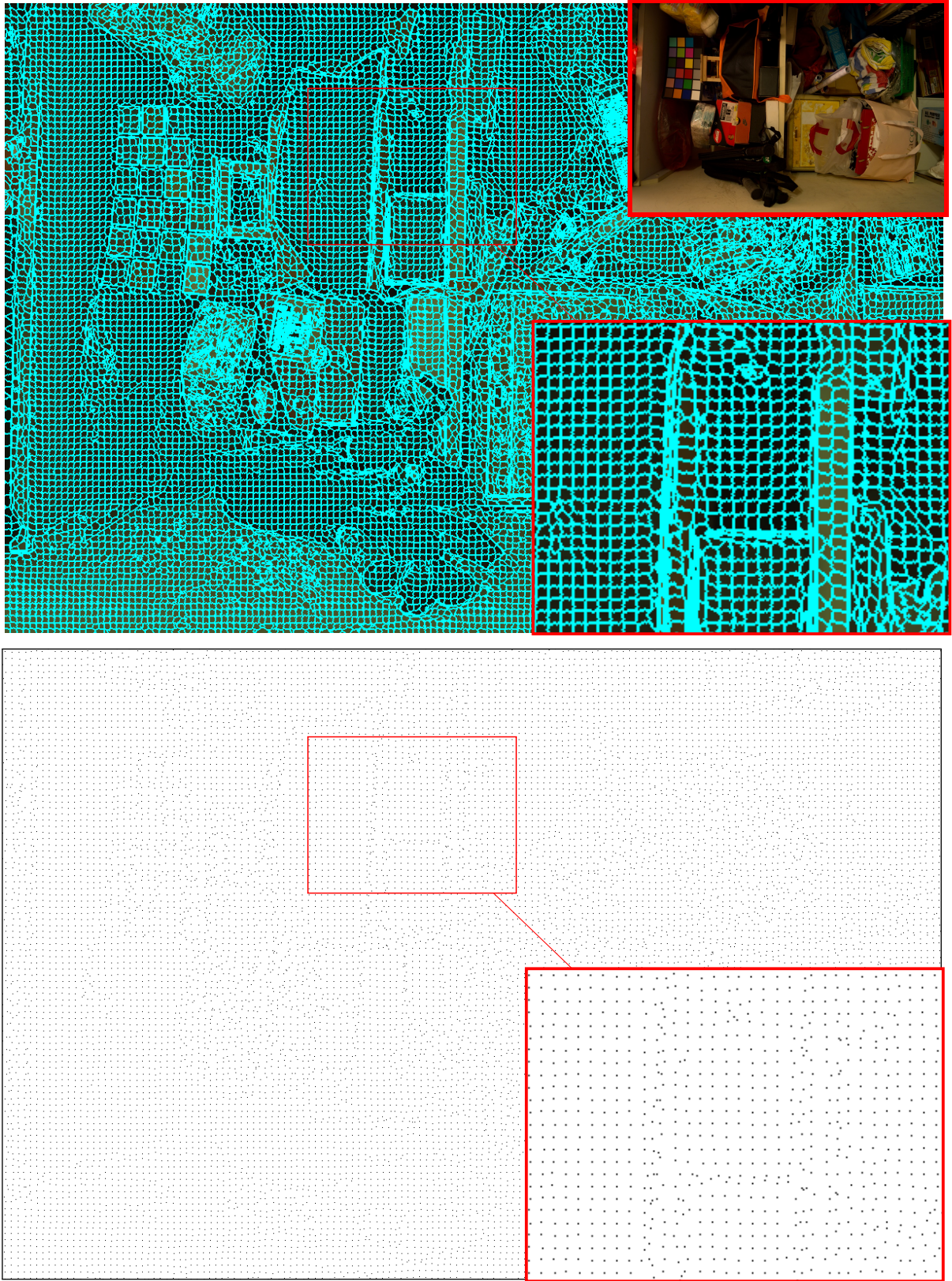Figure A5. Visualization of learned superpixels and sampling mask.
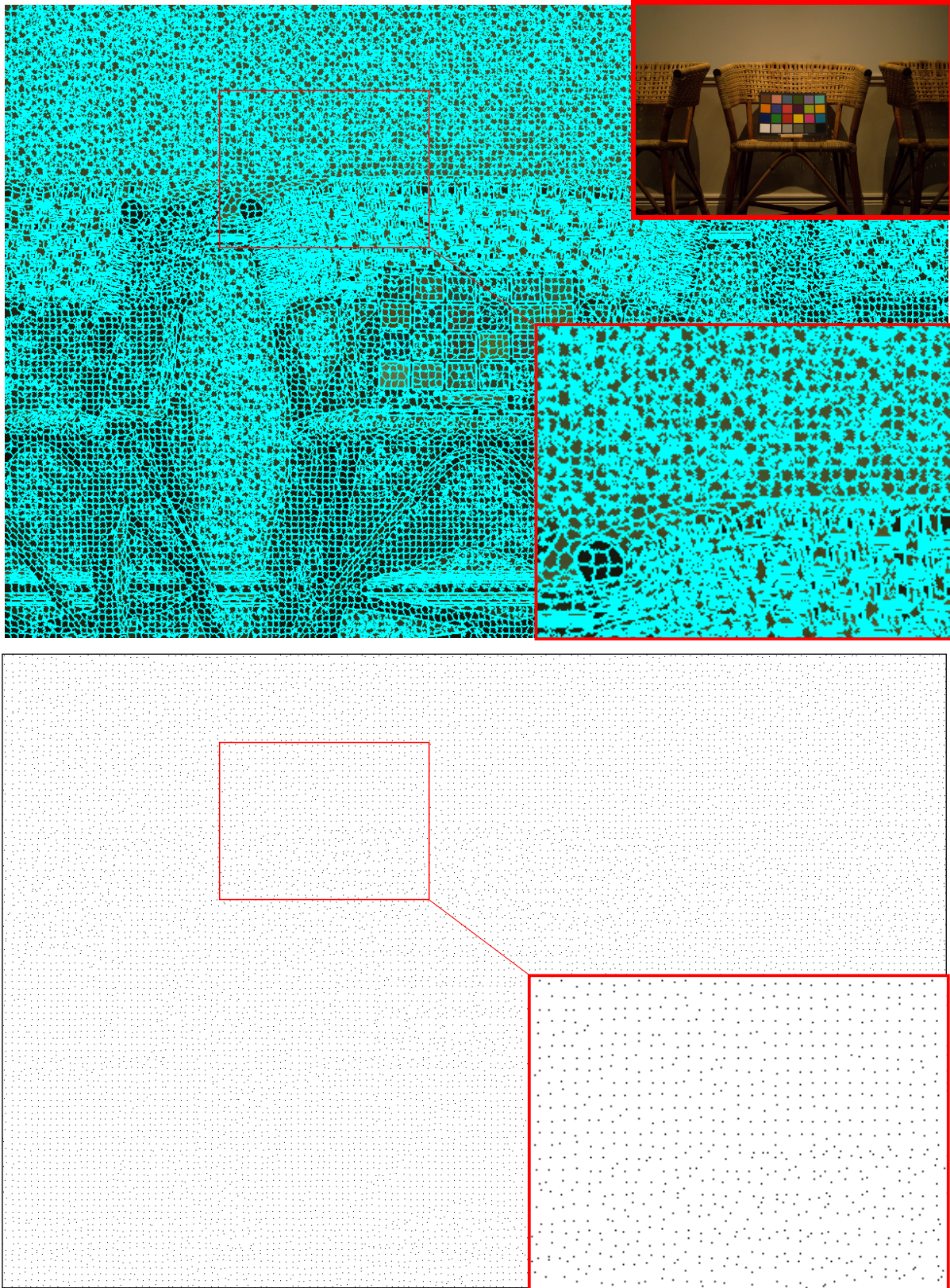
Figure A6. Visualization of learned superpixels and sampling mask.

Figure A7. Visualization of learned superpixels and sampling mask.