RAW Image Reconstruction using a Self-Contained sRGB-JPEG Image with only 64 KB Overhead

Rang M. H. Nguyen   Michael S. Brown
School of Computing, National University of Singapore
nguyenho@comp.nus.edu.sg | brown@comp.nus.edu.sg

Abstract

Most camera images are saved as 8-bit standard RGB (sRGB) compressed JPEGs. Even when JPEG compression is set to its highest quality, the encoded sRGB image has been significantly processed in terms of color and tone manipulation. This makes sRGB-JPEG images undesirable for many computer vision tasks that assume a direct relationship between pixel values and incoming light. For such applications, the RAW image format is preferred, as RAW represents a minimally processed, sensor-specific RGB image with higher dynamic range that is linear with respect to scene radiance. The drawback with RAW images, however, is that they require large amounts of storage and are not well-supported by many imaging applications. To address this issue, we present a method to encode the necessary metadata within an sRGB image to reconstruct a high-quality RAW image. Our approach requires no calibration of the camera and can reconstruct the original RAW to within 0.3% error with only a 64 KB overhead for the additional data. More importantly, our output is a fully self-contained 100% compliant sRGB-JPEG file that can be used as-is, not affecting any existing image workflow - the RAW image can be extracted when needed, or ignored otherwise. We detail our approach and show its effectiveness against competing strategies.

1. Introduction

The vast majority of images used in computer vision and image processing applications are 8-bit standard RGB (sRGB) images, typically saved using the JPEG compression standard. Virtually all image processing application workflows support sRGB and JPEG images. There are many drawbacks, however, when working with sRGB images, e.g. it is well known that sRGB images have a number of non-linear operations applied that makes it difficult to relate the sRGB values back to scene radiance [3, 8, 13, 17, 22, 24].
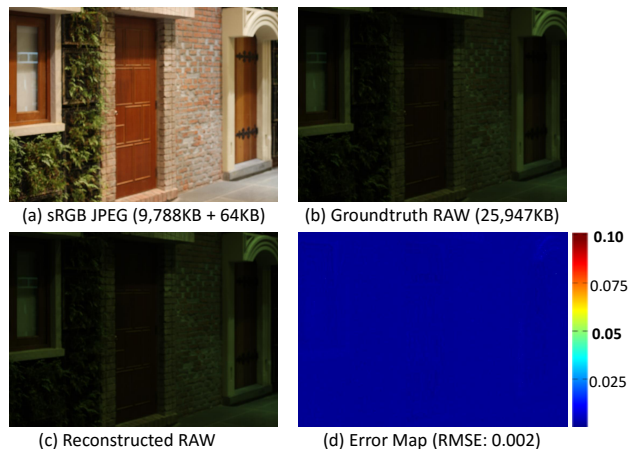

Figure 1. (a) A 5616×3744 resolution high-quality sRGB-JPEG with our metadata embedded (original JPEG size of 9,788 KB, new size of 9,852 KB). (b) Original RAW image is 25,947 KB. (c) Our reconstructed RAW image using the data in the self-contained JPEG. (d) Error map between (b) and (c). Overall reconstruction error is 0.2%.

Most cameras now allow images to be saved in a RAW image format that is an uncompressed, minimally processed image representing the response from the camera sensor. RAW has many advantages over sRGB, including linear response to scene radiance, wider color gamut, and higher dynamic range (generally $12 - 14$ bits). Not surprisingly, RAW is desirable for many computer vision applications, such as photometric stereo, image restoration (e.g. deblurring), white-balance, and more. RAW is also preferred by photographers as it allows the most flexibility in post-processing image manipulation. One of the major drawbacks is that RAW files takes up significantly more space than their sRGB counterpart. In addition, the vast majority of all existing image-based applications are designed to work with 8-bit sRGB images. Images saved in RAW must typically undergo some intermediate process to convert them into sRGB to be useful for many existing tasks.

Given the utility of RAW image data, there has been a number of approaches to map sRGB images back to the

their RAW values. Work by Yuan and Sun [31] demonstrated an effective hybrid-image method that stored a lower resolution version of the original RAW image (e.g. $\frac{1}{2}$ or $\frac{1}{4}$ resolution) and applied smart upsampling that leveraged the sRGB image. One impetus for [31] is that many cameras now support a small-RAW format that save the RAW image in either half and quarter-size resolutions. However, it is important to note that these smaller RAW images still require roughly $1.5 - 6$ MB to store. Other closely related work [30, 3, 17, 19, 20] used a calibration procedure to compute parameters to model the onboard camera processing pipeline in order to reserve sRGB values back to their RAW values. While this calibrated metadata is generally smaller than the $1.5 - 6$ MB needed using the small-RAW strategy, these methods still have a drawback that additional metadata needs to be stored separately for reconstructing the RAW image. Moreover, the calibration procedure needs to be done for several different settings on the camera. The goal of this paper is to provide a fully self-contained JPEG image that allows RAW image reconstruction. In addition, we want to do this with a small memory overhead and in a manner that is 100% compatible with existing JPEG standards. Figure 1 shows an example of our proposed method's ability. An sRGB-JPEG image is embedded with 64 KB of metadata that is used to reconstruct the RAW image with an overall reconstruction error of less than 0.3% (in RAW pixels values).

Contribution We provide a straight-forward and effective procedure to extract the necessary data for reconstructing a RAW image given the corresponding sRGB-JPEG image. As part of this procedure, we describe a fast breadth-first-search octree algorithm for finding the necessary control points to provide a mapping between the sRGB and RAW sensor color spaces. In addition, we also describe a method to encode our data efficiently within the allowed 64 KB text-comment field that is supported by the JPEG compression standard. This allows our method to be fully compatible with existing JPEG libraries and workflows. We compare our approach with existing methods and demonstrate the usefulness of the reconstructed RAW on two applications: white-balance correction and image-deblurring.

2. Related Work

Work related to RAW image reconstruction can be categorized into two areas: radiometric/camera color calibration and methods for image-upsampling.

Radiometric/Color Calibration are methods that aims to compute the necessary mappings to invert the non-linear transformations applied onboard cameras in order to have pixel values that are linear with respect to scene radiance. Conventional radiometric calibration algorithms used multiple images taken with controlled exposures in order to compute inverse response functions of the camera output in-

tensity values to the incoming light. These methods targeted greyscale images [21], or computed a individual response functions per color channel [8, 13, 22, 24]. The main difference among these methods are the models used to represent the response function, e.g. exponentiation [22], polynomial [24], non-parametric [8], and PCA-based model [13].

These early methods discarded RGB values that were too saturated, treating them as outliers to the radiometric model. Work in [2] and [17] found that these outliers were due to limitations in the radiometric models being used. To overcome this, Chakrabarti et al. [2] proposed a method that used combinations of cross-channel linear transforms with per-channel multi-variate polynomials to model the camera color mapping process. Kim et al. [17] proposed a in-camera imaging model that introduced an additional gamut mapping step for handling the out-of-gamut (i.e. saturated) colors. Later, Chakrabarti et al. [3] extended this idea and suggested using uncertainty modelling for handling the quantization of the sRGB colors. These methods significantly improved the ability to reverse sRGB images back to their RAW values, however, they do have two limitations with respect to the problem addressed in this paper. The first limitation is the need to calibrate the color models for a given camera. As discussed by [17], this involves computing multiple parameterized models for different camera settings (e.g. different picture styles). As a result, a single camera would have several different color mappings. Such calibration can be burdensome in practice. The second drawback is the parameterized models are still saved as offline data and the appropriate model based on the camera settings needs to be determined when one desires to reverse an sRGB image.

Image Upsampling are methods that attempt to increase the resolution, or quality, of an image. Representative work include interpolation-based methods [15, 29], edge-based methods [6, 9, 27], and example-based methods [11, 12]. These methods leverage a dictionary of image patches from high-quality images that are used to guide the upsampling process. The most similar to the problem addressed in this paper is the work by Yuan and Sun [31] who demonstrated a hybrid-image method that stored a low resolution version of the RAW image. The low resolution RAW image was upsampled to have the same resolution as the sRGB image by using the sRGB image to guide the upsampling process. The RAW images used in this work were one half or one quarter size of the original RAW image. While these small-RAW are smaller than the original RAW image, they are still require approximately $1.5 - 6$ MB to store. Also, like the work of [17] and [3], this approach requires additional data to be stored separately from the JPEG image in order to perform upsampling.

The work in this paper is distinguished from prior work in two ways. First, the aim here is to embed the neces-
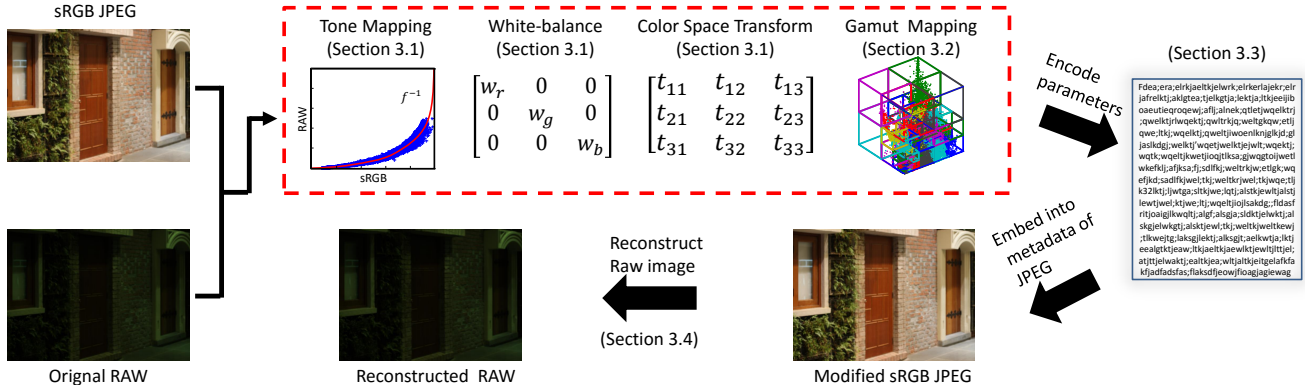
Figure 2. This figure shows an overview of our approach. The section of the paper detailing each component is shown.

	Orignal Size	Type	Storage Size (byte)
T_w^{-1}	3×1	double	24
T_s^{-1}	3×3	double	72
f^{-1}	256×1	int16	512
g^{-1}	4728×6	int16	$56,736$

Table 1. This table shows the amount of data allocated to model the camera-pipeline parameters for a RAW-sRGB image pair. The goal is to keep this within 64KB. The g^{-1} allows up to 4728 control points pairs consisting of an sRGB and RAW-rgb color point (i.e. 6 values in total).

sary information for RAW reconstruction inside the existing sRGB-JPEG image. This is more practical than requiring the user to maintain a companion file containing the necessary data for reconstruction (e.g. small-RAW file or camera calibration data). Second, unlike the radiometric approaches, our approach does not require a calibration procedure involving multiple images. Instead, we only need to estimate a mapping between the given sRGB and RAW image pair provided by the camera. To this end, our goal is to efficiently estimate this necessary metadata such that it can fit inside a single 64 KB text-based comment that can be embedded in the JPEG image [1, 14].

3. Proposed Approach

The work by Kim et al. [17] and Chakrabarti et al. [3] have shown the processing from RAW to sRGB can be described by a number of parameters that model various steps in the onboard camera color processing pipeline. In particular, there is a white-balance matrix T_w, and a color correction matrix T_s, that is first applied. The color correction matrix is used to convert the camera-specific RAW color space into a device-independent color space. This is followed by a tone map operator f applied to all color channels and a gamut mapping g that maps an 3-D color value to a new 3-D color value. The work in [17, 3] parameterized these mod-

els by using many sRGB-JPEG pairs of color charts under different illuminations captured by the camera with specific settings. This type of parameterization is necessary when the input is an arbitrary sRGB image. We follow a similar procedure, but our problem is slightly simplified as we only need to compute these parameters for a single pair of RAW image E and sRGB-JPEG image I. In addition, our approach needs to keep the results to within 64 KB overhead and embed this as a text field in the JPEG image.

Figure 2 provides a diagram that illustrates the overall procedure of our method. Our input is the RAW image captured from the camera and the corresponding sRGB-JPEG. We assume the RAW image has been demosaiced and use the DCRAW utility [5] to perform this task. The data storage budget for each part of the camera model is pre-allocated as shown in Table 1. The total budget is less than 64 KB because it will later be converted to a text format that avoids a `0x00` bit sequence (described in Section 3.3). The following sections describe how the metadata is computed and embedded in the sRGB-JPEG file.

3.1. In-Camera Imaging Model Estimation

Our first step is to compute an inverse tone-curve, f^{-1}, from a pair of sRGB-JPEG and RAW images. This is used to make the sRGB values more linear with respect to the RAW value. We assume that the color correction matrix did not change the brightness of the RAW colors. Additionally, it is assumed that the gamut-map targets chromatic colors, e.g. $g(\mathbf{p}) = \mathbf{p'}$ where $\mathbf{p}$ is an achromatic color (i.e. sRGB saturation value less than 0.2). Based on these two assumptions, the images are converted into the HSV color space, and the V channels are selected to estimate the inverse tone-curve f^{-1}. This curve can be estimated using the spline fitting technique [26] as follows:

$$\frac{1}{N}\sum_{i=1}^{N}||f^{-1}(I_i) - E_i||^2 + \lambda||\nabla^2 f^{-1}||^2,$$
$$\text{s.t } \nabla^1 f^{-1} \geq 0 \tag{1}$$

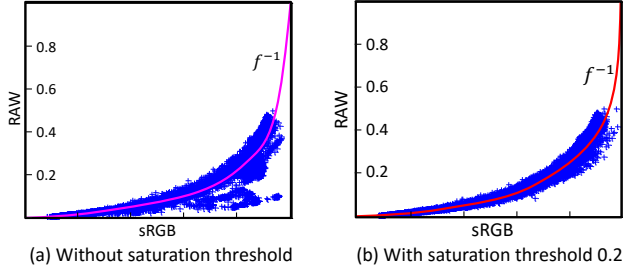(a) Without saturation threshold   (b) With saturation threshold 0.2

Figure 3. This figure shows an example of with/without using saturation threshold for estimating an inverse tone-curve f^{-1}.

where i is the index to color pixels, N is the number of color points after applying a saturation threshold, $\bigtriangledown^1$ denotes the first derivative, and $\bigtriangledown^2$ denotes the second derivative. The first term measures the agreement of f^{-1} with the observations while the second term constraints the smoothness of the curve. The weight λ controls the relative contribution of the two ($\lambda = 0.1$ for all our examples). There is also a hard constraint that the function is monotonically increasing. Note that color values with any channel set to 255 are not used as they represent fully saturated pixels. Figure 3 shows an example of with/without using the saturation threshold for estimating an inverse tone-curve f^{-1}.

After the tone mapping f is estimated, sRGB values are converted to linearized sRGB values using f^{-1}. As with the tone-curve, the linear color correction matrix T_c is computed using the color values with low color saturation that are not affected by the gamut mapping. Here, the color correction matrix T_c is the combination of the white balance matrix T_w and the color space transformation matrix T_s. We estimate the matrix T_c that minimize the following error function:

$$\sum_{i=1}^{N} ||f^{-1}(I_i) - T_c E_i||^2. \qquad (2)$$

Note that most of consumer cameras that are supported RAW format often embeds the white-balance matrix T_w with the RAW files. With T_w, we can obtain the color correction matrix T_s from T_c ($T_c = T_s \times T_w$). Decomposing the color correction matrix into two matrices: white-balance and the color space transformation (T_s) will provide several advantages for editing tasks such as the white-balance modification.

According to [17], the color gamut mapping g can potentially be highly non-linear and challenging to model using parametric functions. We therefore use scattered point interpolation to model this mapping as done in [17]. These scattered points are sampled from the input image. We examined three different strategies to select the scattered points, namely: uniform sampling, k-means clustering, and octree partitioning. The mean values for each partition or cluster
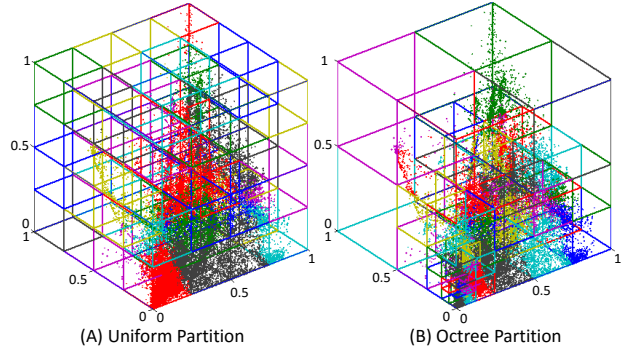

(A) Uniform Partition   (B) Octree Partition

Figure 4. This figure shows an example of partition color space using uniform and octree approaches. The same number of bins $64 = 4^3$ is used for both two approaches.

are chosen as the control points.

It is worth noting that the sRGB and RAW colors in an image are rarely distributed through the whole color space. Therefore, using uniform sampling for the color space has two disadvantages. The first is that many samples are empty and need to be ignored. This makes it hard to know the number of non-empty samples before hand and therefore challenging to efficiently control the exact number of non-empty samples. Second, the number of colors are not distributed equally, and non-empty samples may not represent a good usage of allocating control points. Figure 4-(a) shows an example using uniform sampling. These also means lattice-based methods (e.g. [20]) are also not the right fit for this problem as they attempt to define a lattice over the entire color space. A straight-forward solution would be to use k-means clustering. The drawback for k-means clustering is the required running-time since the number of points and clusters are relative large (e.g. 10^6 in the RAW image). In this paper, we adapt an octree algorithm proposed by Meagher [23] to partition the RGB color space. To do so, we introduce a slight modification to the traditional octree approach that is based on a depth-first mechanism, to one that uses a breadth-first approach. This breadth-first octree construction allows us to control the number of non-empty partitions and sample more scatter points in dense regions (as shown in Figure 4-(b)). The details of the octree implementation is presented in the next section.

3.2. Modified Octree Partitioning

The basic approach for octree partitioning is a depth-first search that explores as far as possible along each branch before backtracking. The implementation starts with a single bin surrounding all 3D (R/G/B) input points. This bin is recursively subdivided into eight smaller bins. Recursion is stopped when a certain criteria is met. Three common conditions to stop bin subdivision are: 1) a bin contains fewer than a given number of points δ_p; 2) a bin reaches the minimum volume δ_l; 3) a bin reaches the maximum depth δ_d.

	Uniform	K-means	Octree
Begin	4096	4096	4096
Return	283	4096	4091
Time(s)	0.53	101.70	11.18
RMSE	0.0037	0.0026	**0.0023**

Table 2. This table shows the three different strategies to select the scattered points for modeling the gamut mapping. These are uniform partition, k-means clustering, and our octree partitioning.

However, in our situation, since we need to limit the amount of control points estimated, we need to keep track of the number of non-empty bins and stop the partition process when it reaches a given number of bins. Using the depth-first search strategy, some of the tree nodes may be divided many times while the others may not be divided although they are similar size and contain similar the number of points. To overcome this limitation, we modify the octree partitioning to use a breath-first search strategy. This involves using a queue Q to store all the current leaf-nodes. At each iteration, the node at the front of the queue Q is extracted and checked whether it satisfies one of the above stopping conditions or not. If the nodes need further division, the non-empty sub-bins of its will be added to the rear of the queue Q and the number of non-empty bins is updated. This process will be iterated until it reaches the desired number of bins K. By doing so, bins having similar size and the number of points will be divided a similar number of times. The details for our modified octree partition are shown in Algorithm 1.

Table 2 shows the comparison among the three strategies to select the scattered points for modeling the gamut mapping function g. Here, the same number of bins ($4096 = 16^3$) is used in all three sampling methods. As can be seen, the running time for uniform sampling is the smallest but its reconstructed errors are highest since the number of non-empty bins is relatively small (around 7%). Using k-means can obtain reasonable reconstructed results but the running time is significantly high. Our modified octree sampling obtains the best reconstructed results and is $10\times$ faster than k-means clustering. It is worth noting that the octree partitioning may not guarantee the exact number of returned non-empty bins as one or more sub-bin(s) (up to eight) are created at each division. However, it can reach very close to the given number within 8 control points.

3.3. Metadata Embedding

After color mapping parameters are estimated, they are embedded as metadata into the JPEG file. The metadata structure in JPEG contains several segments. Each segment has capacity for a different kind of data. These are delimited by two-byte codes called markers [1, 14]. One of these segments is the comment (COM) segment. A COM seg-

Algorithm 1 Modified Octree Partition

Input: a set of n color points $\{p_i\}_{i=1}^n$, the desired number of bins K, the minimum capacity δ_p, the minimum size of bin δ_l, and the maximum depth of bin δ_d.

1: $B \leftarrow [n]$ ▷ Array of Point Counts
2: $C \leftarrow [\min(p_i), \max(p_i)]$ ▷ Array of Bin Conners
3: $D \leftarrow [0]$ ▷ Array of Bin Depths
4: $M_i \leftarrow 1$ ▷ Map of Point Bin
5: create an empty queue Q
6: $Q.enqueue(1)$ ▷ Insert the first bin
7: $t \leftarrow 1$ ▷ Total number of non-empty bins
8: **while** $(t < K) \wedge \neg Q.isEmpty()$ **do**
9:     $u \leftarrow Q.dequeue()$
10:    $L_u \leftarrow \min(C_u(4:6) - C_u(1:3))$
11:    $scd \leftarrow (B_u < \delta_p) \vee (L_u < \delta_l) \vee (D_u > \delta_d)$
12:    **if** scd **then**
13:       continue   ▷ Move to the next bin
14:    **end if**
15:    $m \leftarrow (C_u(1:3) + C_u(4:6))/2$ ▷ The center
16:    $t \leftarrow t - 1$ ▷ Remove the old bin
17:    **for** $i = 1:8$ **do**
18:       $v \leftarrow length(D) + 1$ ▷ New bin number
19:       $D_v \leftarrow D_u + 1$ ▷ Increase depth
20:       $C_v \leftarrow getConners(m, C_u)$
21:       Calculate $mask$ which points belong in v
22:       $M_{mask} \leftarrow v$
23:       $B_v \leftarrow countPointNumber(M, v)$
24:       **if** $B_v > 0$ **then**
25:          $t \leftarrow t + 1$ ▷ Insert the new bin
26:          $Q.enqueue(v)$
27:       **end if**
28:    **end for**
29: **end while**

Output: a map of point bins M.

ment does not interfere with the image stored in the JPEG file. The maximum size of a COM segment is 64 KB.

We used the COM segment to store estimated color mapping data. Since the COM segment can only contain a text string, we need to encode these parameters into an array of characters, however, we must avoid the special character "null" (i.e. 0×00) as it denotes the end of the text comment. To avoid the null character in the sequence of characters, we used a simple (and fast) scheme as follows. The character sequence is converted into a sequence of binary bits. At every seventh bits, an additional bit 1 is inserted. This new bit stream is then converted to ASCII characters. By inserting this additional bit in this periodic manner the COM segment will not contain the null character. Figure 5 shows an example of this approach. By inserting this additional bit, the real storage size of metadata in the COM segment reduces to 56 KB ($7/8$ of the original size 64 KB). This is why the data allocation shown in Table 1 is only 56 KB.
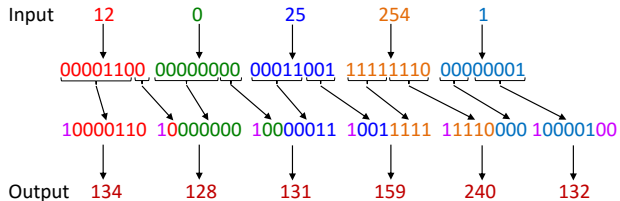
Input	12	0	25	254	1
	00001100	00000000	00011001	11111110	00000001
	10000110	10000000	10000011	10011111	11110000 10000100
Output	134	128	131	159	240 132

Figure 5. This figure shows an example of our encoding method which avoids null characters.

Camera Name	Ours	[31] (1/2)	[31] (100 × 90)
Canon 1Ds	**0.0018**	0.0049	0.0135
Canon 600D	**0.0038**	0.0085	0.0191
Nikon D5200	**0.0033**	0.0078	0.0173
Sony α57	**0.0020**	0.0055	0.0150

Table 3. This table shows the comparison between our method and up-sampling method proposed in [31] in terms of RMSE. For up-sampling method [31], RAW images at resolutions of $1/2$ of the original size and 100×90 are used for upsampling.

3.4. RAW Reconstruction

To reconstruct the RAW values, the metadata embedded in the JPEG file is first extracted and decoded by converting the text string to an bit string and then removing the additional bit pattern. We now have back all the parameterized data: the inverse white-balance matrix T_w^{-1}, the inverse color space transformation T_c^{-1}, the control points for the inverse gamut map g^{-1} and the control points for the inverse tone mapping f^{-1}. The RAW values are reconstructed by first applying the inverse tone-mapping f^{-1} to obtain the linearized sRGB image. Then the gamut mapping is applied. We adopt a linear tetrahedral interpolation [16] to model the gamut mapping since the scattered points are in 3D color space. Next, the inverse color space transformation is applied and finally, the white-balance step is undone to obtain the reconstructed RAW image.

4. Experiments

In this section, we compared our RAW reconstruction strategy with alternative techniques, including our method using k-means clustering, our method using the octree, and the upsampling method by Yuan and Sun [31]. Yuan and Sun [31] do not provide code and we have re-implemented their approach. We note that we have modified their approach to use the recent state-of-the-art method proposed by Ferstl et al. [10] to perform the image upsampling. Images used in this paper were taken from various online data sets that contain RAW-sRGB pairs, including images from [17], [4], [25], and [7].

Figure 6 shows the results for images from various cameras and scenes. A jet map is used to show the error between the original and reconstructed RAW image. The root mean square error (RMSE) is also shown to quantitatively evaluate the results. For Yuan and Sun's method [31], RAW images at resolutions of $1/2$ of the original size are used for upsampling. For a fairer comparison, we also used a small-RAW image of resolution 100×90 which can be placed inside the 64 KB metadata. Table 3 shows the results in terms of RMSE. For each camera, 30 images were examined and the average RMSE was reported. The proposed octree partitioning approach provides RAW reconstructions

with the lowest RMSE. Note that the errors shown also includes quantization errors in the 8-bit sRGB image which is around 0.001, or 0.1% in average.

The proposed method does not attempt to remove compression artifacts that arise due to the lossy nature of JPEG. We assume that the input image is saved as a high-quality JPEG, however, for sake of completeness, we examine the effect of different JPEG qualities on the reconstructed RAW results. Most DSLR cameras support three image quality types, e.g.: fine, normal, and basic, that corresponds to the compression ratios $(1/4, 1/8$ and $1/16)$. Figure 7 shows an example for a Nikon D5200 camera. Unsurprisingly, the quality does affect the reconstructed RAW-RGB images, however the overall difference between them is not significant.

5. Applications

We demonstrate the usefulness of the reconstructed RAW image data with two applications: white-balance and image deblurring. It is well known that having access to the original RAW image is advantageous for these applications.

5.1. White-Balance Correction

As noted in the processing pipeline in Figure 2 in Section 3, white-balancing is a procedure that is applied early in the processing pipeline. Attempting to change white-balance in the sRGB image is challenging as it cannot undo the photo-finishing that has been applied to the sRGB image. In this experiment, we compared applying white-balance correction on our reconstructed RAW and the original sRGB-JPEG. The comparison results are shown in Figure 8. The input images are captured under wrong white-balance settings; while the ground truth images are captured under the proper settings. Here, the achromatic colors on the color checker boards are manually selected to use as the scene illumination color. White-balance correction on the reconstructed RAW images is visually better than correction using the sRGB images.

5.2. Image Deblurring

Image deblurring assumes a linear image formation model in the form: $I_B = I \otimes h$, where I is the latent im-
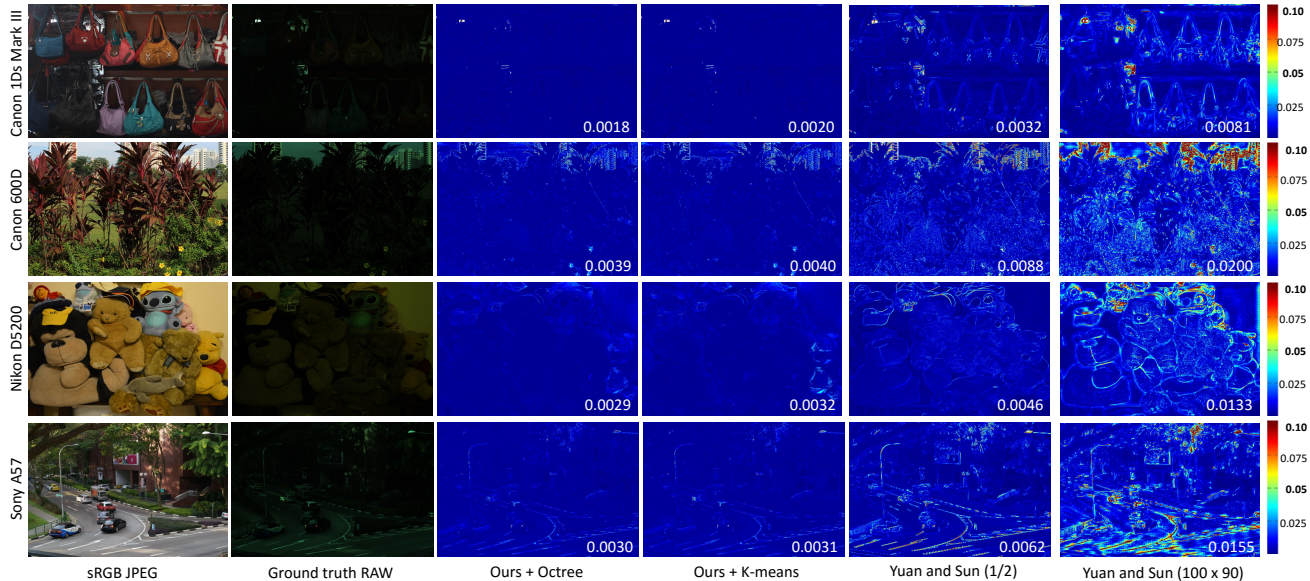
Figure 6. This figure shows comparisons between our approach and our implementation of the upsampling approach proposed by Yuan and Sun [31] for various scenes and cameras (a Canon 1Ds Mark III, a Canon 600D, a Nikon D5200, and a Sony α57). The white points on the difference maps indicate overexposed pixels with a value of 255 in any of the channels. The RMSEs for the each method are shown in the bottom right of each error map.
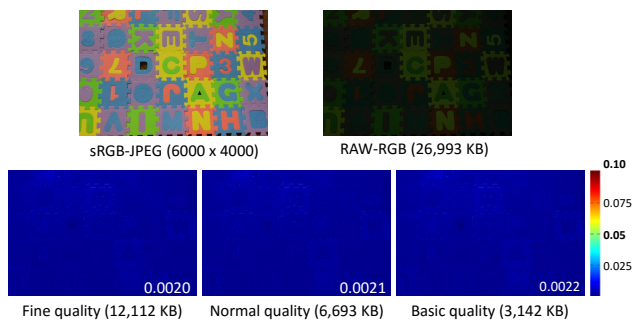

Figure 7. This figure shows an example of using different qualities of sRGB-JPEG images for reconstructing the RAW-RGB image. Here, three different qualities: fine, normal, and basic (which supports in Nikon cameras) are examined. The RMSEs for the each quality are shown in the bottom right of each error map.

age and h is a blur kernel. For blurred sRGB images, the relationship is not truly linear between I_B and I. Work by Tai et al. [28] showed that the non-linear response of the camera changes the shape of the h to a spatially varying convolution making image deblurring even more challenging. Thus, it is desirable to deblur the image in the linear RAW space. We compared the deblurring method proposed in [18] on our reconstructed RAW and sRGB-JPEG images. This is done by applying a motion blur on a ground truth RAW image and then use the estimated parameters in the camera color pipeline to synthesize the blurred sRGB input images. Figure 9 shows the results of the deblurred sRGB and deblurred reconstructed RAW image. The signal-to-

noise ratios (SNRs) are also reported at the bottom right of each image. Deblurring of the reconstructed RAW images gives superior results.

6. Discussion and Conclusion

We have described a method to encode the necessary metadata with the sRGB image for reconstructing a high-quality RAW image. Our approach produces a fully self-contained 100% compliant JPEG file that can be used as-is, not affecting any existing image workflows. This method can reconstruct the original RAW to within 0.3% error with only 64 KB overhead to the original JPEG file.

One drawback of our method is that we cannot properly handle sRGB image values that are saturated or images that have spatially-varying tone-mapping applied. Upsampling methods (e.g. [31]) can better estimate these values given the availability of spatial RAW values in these regions. Future work would be to embedded additional RAW values spatially to help in-paint or interpolate these regions. We also note that we have optimized our framework to minimize error for backward mapping from sRGB to RAW, however, for many photography tasks (such as our white-balance example), the forward mapping from RAW back to sRGB is needed. A future topic would be to modify our method to consider this bi-directional reconstruction error.

Figure 8. This figure shows examples on correcting white-balance for different cameras: a Canon 1Ds Mark III, a Nikon D5200, and a Sony α57. The first column is the input images captured under the wrong white-balance settings; the second column shows the ground truth images captured under the proper settings. The third column displays the results applied the white-balance correction on our reconstructed RAW images. The final column shows the results applied the white-balance correction directly on the sRGB-JPEG images.
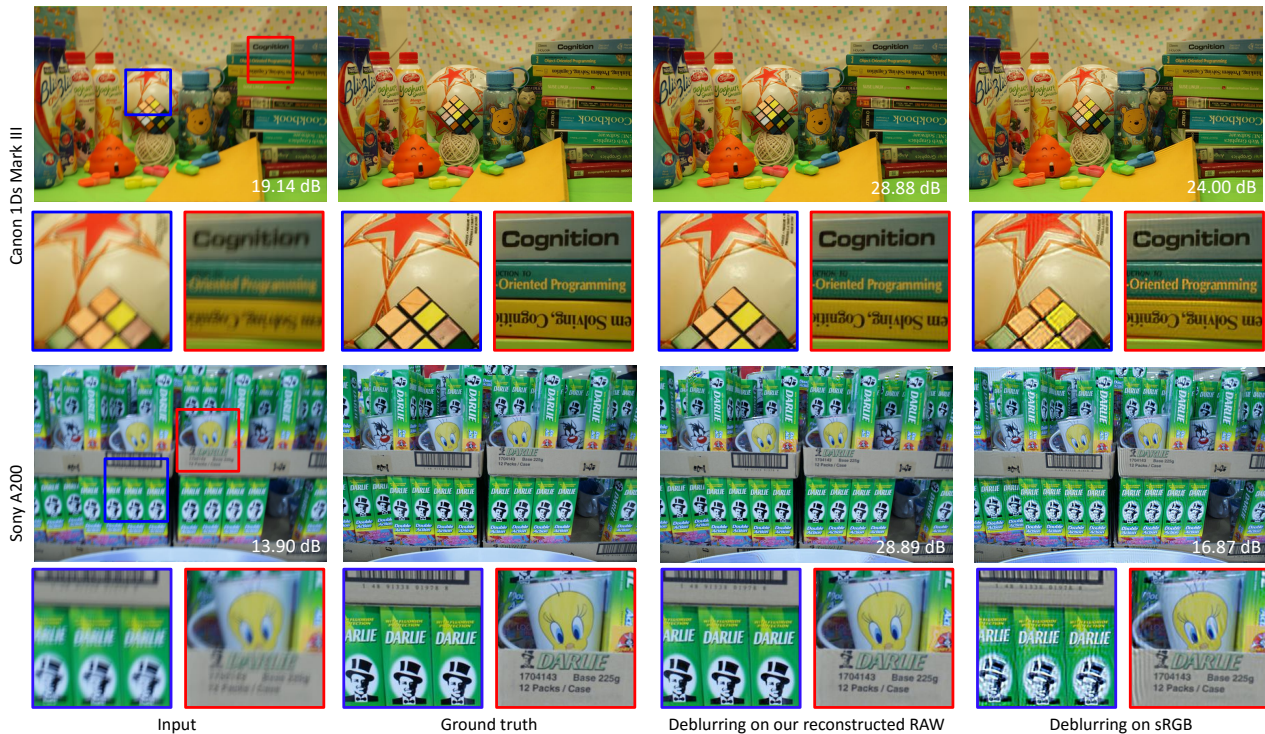

Figure 9. This figure shows examples for image deblurring. A motion blur on the non-blurred ground truth RAW images is performed. The blurred sRGB image is synthesized using the parameterized color pipeline model. We applied our method to reconstruct the blurred RAW image, then deblurred it, and converted it back to the sRGB image. The first and third rows show the results, while the second and fourth rows shows close-ups of particular regions. The signal-to-noise ratios (SNRs) were reported at the bottom right of each image.

References

[1] B. Brower, A. Hinds, E. Hamilton, D. Lee, and G. Sullivan. Information Technology: Digital compression and coding of continuous-tone still images: JPEG File Interchange Format (JFIF). *ISO/IEC*, 2011.

[2] A. Chakrabarti, D. Scharstein, and T. Zickler. An empirical camera model for internet color vision. In *British Machine Vision Conference (BMVC)*, 2009.

[3] A. Chakrabarti, Y. Xiong, B. Sun, T. Darrell, D. Scharstein, T. Zickler, and K. Saenko. Modeling radiometric uncertainty for vision with tone-mapped color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2014.

[4] D. Cheng, D. K. Prasad, and M. S. Brown. Illuminant estimation for color constancy: why spatial-domain methods work and the role of the color distribution. *Journal of the Optical Society of America A (JOSA A)*, 2014.

[5] D. Coffin. DCRAW: decoding raw digital photos in linux. https://www.cybercom.net/~dcoffin/dcraw/, 1997. Accessed: 2015-10-05.

[6] S. Dai, M. Han, W. Xu, Y. Wu, and Y. Gong. Soft edge smoothness prior for alpha channel super resolution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[7] D.-T. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato. Raise: A raw images dataset for digital image forensics. In *ACM Multimedia Systems Conference*, 2015.

[8] P. E. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *ACM SIGGRAPH 2008 classes*, 2008.

[9] R. Fattal. Image upsampling via imposed edge statistics. *ACM Transactions on Graphics (TOG)*, 2007.

[10] D. Ferstl, C. Reinbacher, R. Ranftl, M. Ruether, and H. Bischof. Image guided depth upsampling using anisotropic total generalized variation. In *IEEE International Conference on Computer Vision (ICCV)*, 2013.

[11] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *International Journal of Computer Vision (IJCV)*, 2000.

[12] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *IEEE International Conference on Computer Vision (ICCV)*, 2009.

[13] M. D. Grossberg and S. K. Nayar. Determining the camera response from images: What is knowable? *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2003.

[14] E. Hamilton. JPEG File Interchange Format. *C-Cube Microsystems*, 1992.

[15] H. S. Hou and H. Andrews. Cubic splines for image interpolation and digital filtering. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1978.

[16] J. M. Kasson, W. Plouffe, and S. I. Nin. Tetrahedral interpolation technique for color space conversion, 1993.

[17] S. J. Kim, H. T. Lin, Z. Lu, S. Susstrunk, S. Lin, and M. S. Brown. A new in-camera imaging model for color computer vision and its application. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012.

[18] D. Krishnan and R. Fergus. Fast image deconvolution using hyper-laplacian priors. In *Advances in Neural Information Processing Systems*, 2009.

[19] H. Lin, S. J. Kim, S. Süsstrunk, and M. S. Brown. Revisiting radiometric calibration for color computer vision. In *IEEE International Conference on Computer Vision (ICCV)*, 2011.

[20] H. T. Lin, Z. Lu, S. J. Kim, and M. S. Brown. Nonuniform lattice regression for modeling the camera imaging pipeline. In *European Conference on Computer Vision (ECCV)*. 2012.

[21] S. Lin and L. Zhang. Determining the radiometric response function from a single grayscale image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[22] Mann, Picard, S. Mann, and R. W. Picard. On being 'undigital' with digital cameras: Extending dynamic range by combining differently exposed pictures. In *Proceedings of IS&T*, 1995.

[23] D. J. Meagher. Octree encoding: A new technique for the representation, manipulation and display of arbitrary 3-d objects by computer. Technical report, Image Processing Laboratory, Rensselaer Polytechnic Institute, 1980.

[24] T. Mitsunaga and S. K. Nayar. Radiometric self calibration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999.

[25] R. M. H. Nguyen, D. K. Prasad, and M. S. Brown. Raw-to-raw: Mapping between image sensor color responses. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[26] C. H. Reinsch. Smoothing by spline functions. *Numerische Mathematik*, 1967.

[27] J. Sun, J. Sun, Z. Xu, and H.-Y. Shum. Image super-resolution using gradient profile prior. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[28] Y.-W. Tai, X. Chen, S. Kim, S. J. Kim, F. Li, J. Yang, J. Yu, Y. Matsushita, and M. Brown. Nonlinear camera response functions and image deblurring: Theoretical analysis and practice. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013.

[29] P. Thévenaz, T. Blu, and M. Unser. Image interpolation and resampling. *Handbook of Medical Imaging, Processing and Analysis*, 2000.

[30] Y. Xiong, K. Saenko, T. Darrell, and T. Zickler. From pixels to physics: Probabilistic color de-rendering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[31] L. Yuan and J. Sun. High quality image reconstruction from RAW and JPEG image pair. In *IEEE International Conference on Computer Vision (ICCV)*, 2011.