Nonuniform Lattice Regression for Modeling the Camera Imaging Pipeline

Hai Ting Lin[1], Zheng Lu[2],
Seon Joo Kim[3], and Michael S. Brown[1]

[1] National University of Singapore
[2] University of Texas at Austin
[3] SUNY Korea

Abstract. We describe a method to construct a sparse lookup table (LUT) that is effective in modeling the camera imaging pipeline that maps a RAW camera values to their sRGB output. This work builds on the recent in-camera color processing model proposed by Kim et al. [1] that included a 3D gamut-mapping function. The major drawback in [1] is the high computational cost of the 3D mapping function that uses radial basis functions (RBF) involving several thousand control points. We show how to construct a LUT using a novel nonuniform lattice regression method that adapts the LUT lattice to better fit the 3D gamut-mapping function. Our method offers not only a performance speedup of an order of magnitude faster than RBF, but also a compact mechanism to describe the imaging pipeline.

1 Introduction

This paper is concerned with the color mapping process that is applied onboard a camera; i.e. how RAW sensor values are mapped to their corresponding standard RGB (sRGB) outputs. This work falls into the broader topic of radiometric calibration which is the process of recovering scene radiance (or sensor irradiance) from image intensities. Radiometric calibration is a well-studied topic in the computer vision field. Representative works include [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. The mapping between scene irradiance to image intensities is well-known to be nonlinear and is generally modeled by a radiometric response function[1] denoted as f. Existing methods used a variety of approaches to determine f (or its inverse), including observing the changes in the image due to the exposure change [2, 3, 4, 5, 6] or lighting change [7, 8], intensity distributions across edges in single images [10, 11], and statistics from a collection of images [9].

While existing methods were overall effective in modeling the color mapping process, some RGB colors could not be mapped well using the conventional model based on per-channel tone-mapping. Recent work by Kim et al. [1] addressed this issue by proposing to add a 3D gamut-mapping function in the imaging pipeline. This new imaging model was shown to be significantly more accurate at

[1] Also referred to as a camera response function or a tone-mapping function.

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part I, LNCS 7572, pp. 556–568, 2012.

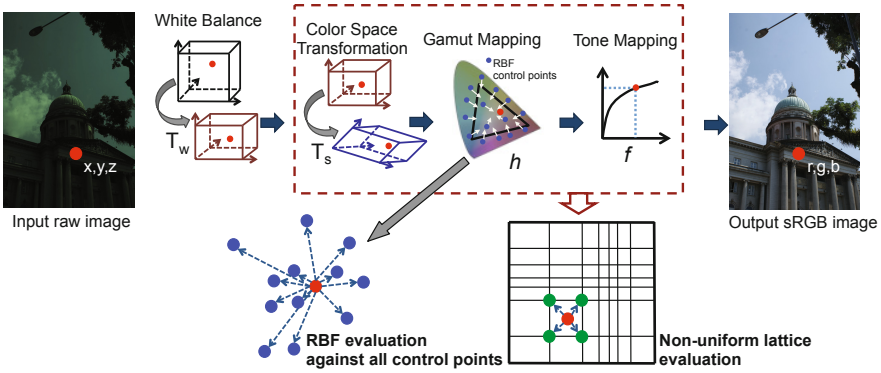

Fig. 1. The in-camera color processing pipeline (adapted from [1]). One significant performance bottleneck is the used of RBF to model the 3D gamut-mapping function. RBF requires computing a distance to all control points in the RBF. Our contribution is to replace several steps in the imaging pipeline with a single sparse LUT based on a nonuniform lattice layout.

modeling the color mapping process than conventional approaches. In addition, the introduction of the color-gamut mapping step made it possible to model different picture styles (e.g. landscape, portrait, vivid, etc). Their model was expressed as:

$$\begin{bmatrix} I_{rx} \\ I_{gx} \\ I_{bx} \end{bmatrix} = \begin{bmatrix} f_r(e_{rx}) \\ f_g(e_{gx}) \\ f_b(e_{bx}) \end{bmatrix}, \text{ where}$$

$$\begin{bmatrix} e_{rx} \\ e_{gx} \\ e_{bx} \end{bmatrix} = h(\mathbf{T}_s \mathbf{T}_w \mathbf{E}_x). \tag{1}$$

In this pipeline, the RAW sensor values $\mathbf{E}_x = [E_{rx}, E_{gx}, E_{bx}]^T$ are first white-balanced by a 3×3 diagonal matrix $\mathbf{T}_w$. Then the white-balanced RAW values, defined in the camera's color space, are transformed to the linear sRGB space by a 3×3 matrix $\mathbf{T}_s$. The color mapping function h ($\mathbb{R}^3 \rightarrow \mathbb{R}^3$) is applied afterwards, followed by a final compression by the radiometric response functions $f_c, c \in \{R, G, B\}$. Figure 1 shows a diagram of this pipeline.

The work in [1] analyzed over 30 different cameras and found that the gamut-mapping function, h, could not be accurately represented by a general parameterizable model or polynomial. Instead, scatter point interpolation via radial basis functions (RBFs) was used to model h. While this was effective, it has a noteable drawback in terms of computational cost because its evaluation requires computing distances to all control points as illustrated in Figure 1. Although a full-resolution look-up table (LUT) was also proposed in [1], their dense LUT required significant memory (over 220MB) and hours to generate. The approach in [1] is inherently deterministic given the input data. Xiong et al. [13] recently proposed a probabilistic method that can potentially improve the accuracy of

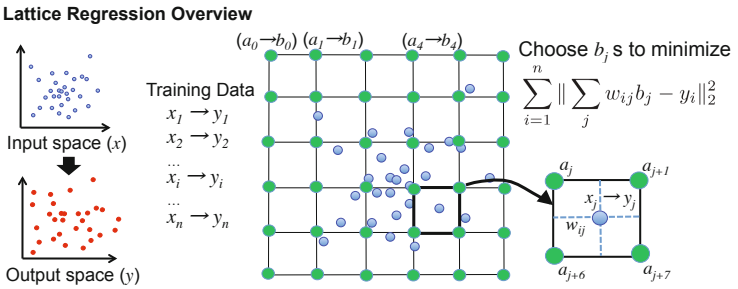

Fig. 2. An overview of lattice regression using uniform node placement. This example is in 2D, but can be easily extended to 3D. The lattice a is defined in the input space. Each lattice node, a_i has a corresponding output value b_i. The idea of lattice regression is to compute the control points b_i that interpolate the training-data based on a given interpolation scheme which define the weights w_{ij}.

the gamut-mapping. Their work use local Gaussian processes [14] to learn the mapping between the RAW and sRGB spaces based on training-data. Similar to RBF, however, the procedure in [14] involves a k-nearest neighbor search which requires the distances to the training-data to be computed.

In this paper, we propose a method to significantly speed up the in-camera color mapping process. Specifically, we introduce a sparse 3D LUT which defines a lattice of control points that are used to interpolate the gamut mapping. This method requires no distance computation, but instead indexes directly to the appropriate LUT cell based on the lattice's structure as illustrated in Figure 1. The general trade-off for using a LUT in lieu of a more complex function is a reduction in accuracy due to the lower-resolution of the control points used for interpolation in the LUT. To address this issue, we have developed an adaptive lattice regression algorithm that modifies the lattice layout in a nonuniform manner to produce a more accurate estimation of the color mapping function. Moreover, we show that our nonuniform lattice regression method is effective enough to combine the color transformations (T_s) and radiometric functions (f_c) into the LUT. Our nonuniform lattice regression method gives performances comparable to the method based on the RBF, but requires a fraction of the time to evaluate.

The remainder of this paper is organized as follows; Section 2 gives a brief overview of the lattice regression and related work; Section 3 describes our nonuniform regression algorithm; Section 4 demonstrates results obtained using our approach followed by the conclusion and discussion in Section 5.

2 Lattice Regression and Related Work

An efficient method to transform between color spaces is to use a sparse LUT [15]. Given a set of RGB colors x in an input color space and a set of points y in

the output color space, a LUT is defined over the input color space as a lattice a, which has vertices $\{a_1, a_2, ..., a_m\}$. This effectively divides the input color space into cells with a_i's as a cell's vertices. The corresponding values of the output color space for the lattice a is denoted as $b = \{b_1, b_2, ..., b_m\}$. The LUT is applied in the following manner. An input point x_i finds its corresponding cell in a and then interpolates the output value $\hat{y}_i$. For linear interpolation, which is commonly used, the output estimation $\hat{y}_i = \sum_{j=1}^{m} w_{ij} b_j$, where w_{ij}'s are the interpolation weights satisfying that $w_{ij} \geq 0$, $w_{ij} = 0$ if a_j is not a vertex of the cell containing x_i, $\sum_{j=1}^{m} w_{ij} a_j = x_i$ and $\sum_{j=1}^{m} w_{ij} = 1$. Figure 2 shows a diagram (drawn in 2D for sake of simplicity) of this type of LUT.

The unknown variables in this approach are the lattice output points b which must be computed based on a training sample pairs $(x_i, y_i), i = 1, 2, .., n$. Assuming a fixed space lattice a and linear interpolation function, the output b can be solved as [16]:

$$\begin{aligned} \hat{b} &= \arg\min_b \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \\ &= \arg\min_b \sum_{i=1}^{n} \| \sum_j w_{ij} b_j - y_i \|_2^2 \\ &= \arg\min_b \|Wb - y\|_2^2. \end{aligned} \tag{2}$$

where W is the weighting matrix with its i-th row being $[w_{i1}, w_{i2}, ..., w_{im}]$ and $y = [y_1, y_2, ..., y_n]^T$.

This basic approach assumes that the training-set is well distributed over the input space. Garcia and Gupta [15, 17] suggested adding a second-order regularization on b to achieve a smoother extrapolation when some cells contained limited number of training samples.

Their proposed objective becomes [17]:

$$\hat{b} = \arg\min_b \|Wb - y\|_2^2 + \lambda J_K(b), \tag{3}$$

where $J_K(b) = b^T K b$ is the thin-plate regularizer with a $m \times m$ matrix K that depends only on the type of interpolation basis function and lattice dimensions, and λ is a weighting scalar.

The easiest design is to distributed a_i's equally in each dimension of the input color space. While distributing a_i equally enables a simple array-style access to the cell needed to perform interpolation, it may waste vertex quota in the region where the transformation is quite flat or allocate too few vertices in regions where the transformation is complicated.

Nonuniform sampling strategies have been introduced. Most notable are Chang et al. [18] and Monga et al. [19]. Although totally free placement of the latex vertices enables the possibility of using less vertices, extra time is introduced during evaluation for sub-volume access [20, 21]. Other approaches such as [18] tried to resolve the sub-volume access using constrained vertices placement. However,

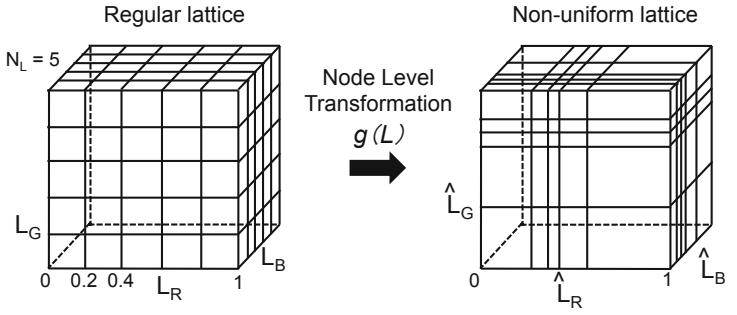

Fig. 3. Our algorithm computes a node level transformation function that transforms a regular lattice to a non-uniform lattice by moving the node levels along different dimensions

these approaches required the space transformation function or its high resolution approximation to be known. In our case, only a sparse set of sample points are known. Moreover, none of the above approaches mentioned can be easily regularized, which is important in our setting.

We propose our own simple, but effective, nonuniform lattice regression scheme based on the objective of reducing fitting error.

3 Nonuniform Lattice Regression

In our formulation, we use the term, *node level*, to denote how the lattice dimensions are divided given the number of lattice nodes N_L along each dimension. We can think of each node level as being a grid line in the lattice structure. Since the input and output spaces are normalized, the node levels for a given dimension spread out in the range $[0, 1]$. For example, if N_L is 6, the node levels, u_i, for a uniform lattice for a dimension will be $0, 0.2, 0.4, 0.6, 0.8$, and 1. In our approach, N_L is the same for all three color channels., however, the algorithm can be easily extended to have different N_L for different channels.

The crux of our approach is to construct a node level transformation function, g, using the smoothed error histogram of the sample points computed based on the uniform lattice. For dimension c, the node level transformation function g_c transforms an input node level u_i to another level along the lattice dimension, i.e. we are determining how to adjust the lattice grids to reduce the overall interpolation error of the lattice. In essence, the function g_c maps the uniform lattice node levels to the non-uniform lattice node levels (see Figure 3). The details of our formulation are described in the following.

Uniform Lattice Initialization
Our algorithm first initializes a uniform lattice using the following equation:

$$\hat{b} = \arg\min_b \|Wb - y\|_2^2 + \lambda_S \|Sb\|_2^2 + \lambda_A \rho(b) + \lambda_K J_K(b), \tag{4}$$

which is extended from Eq. 3 by adding a first-order derivative smoothness regularization $\|Sb\|_2^2$ and a boundary constraint $\rho(b)$. This boundary constraint extension is necessary since in our particular case, data samples are rare in a large portion of the input color space especially near the boundary where the color saturation level is very high. The first-order regularization matrix $S = [S_R; S_G; S_B]$, where each row of $N_L^2(N_L - 2) \times m$ matrix S_c, $c \in \{R, G, B\}$, contains only three nonzero entries. Assuming nodes $a_{i_{1c}}$ and $a_{i_{3c}}$ are neighbors to $a_{i_{2c}}$ along dimension c, there exists one row in S_c with the nonzero values $\frac{-d(i_{3c},i_{2c})}{d(i_{1c},i_{2c})+d(i_{3c},i_{2c})}$, $\frac{-d(i_{1c},i_{2c})}{d(i_{1c},i_{2c})+d(i_{3c},i_{2c})}$ and 1 at corresponding positions respectively, where $d(i, j)$ is the distance between nodes a_i and a_j. The boundary constraint $\rho(b) = \sum_{c\in\{R,G,B\}}((b_1)_c^2 + \sum_{i\in\Upsilon_c}((b_i)_c - 1)^2)$, where Υ_c is the set of a_i nodes whose c coordinates equal to 1. Operator $(\cdot)_c$ extracts the c coordinate of a vector or c column of a matrix. Here we also assume a_1 represents the origin $[0, 0, 0]$. In matrix form, $\rho(b) = \sum_{c\in\{R,G,B\}} \|A_c(b)_c - \xi\|_2^2$, where A_c is composed of standard unit row vectors corresponding to the elements of set Υ_c except the last row being $[1, 0, ..., 0]$ and ξ is vector $[1, ..., 1, 0]^T$. Terms λ_S, λ_A, and λ_K are weighting parameters.

Assuming denotations $\tilde{W}_c = [W; \lambda_S S; \lambda_A A_c]$ and $\tilde{y}_c = [(y)_c; \mathbf{0}; \lambda_A \xi]$, Eq. 4 can be solved in closed form per dimension:

$$\hat{b}_c = (\tilde{W}_c^T \tilde{W}_c + \lambda_K K)^{-1} \tilde{W}_c^T \tilde{y}_c. \tag{5}$$

The matrix K is only constructed once and is used again when constructing the nonuniform lattice. The matrix K only depends on the lattice size and the interpolation function adopted for the smoothness constraint. We select tricubic interpolation for the smoothness constraint to construct K (see [17] for details).

In Eq. 5, the weighting scalars $\lambda_S = 0.1$, $\lambda_A = 0.01$, and $\lambda_K = 1e - 6$ are fixed in all our computations. We denote this initial uniform lattice as a^o and the corresponding output vertex set as b^o, and W^o is the weighting matrix of x in a^o.

Node Level Transformation Formulation

In our formulation, the node level transformation function g_c can be best explained through its inverse function g_c^{-1} that is constructed based on the error histogram computed from the uniform lattice a^o. The idea is that the error histogram indicates locations where the non-uniformity is most prominent and hence more dense lattice nodes are needed.

From the initial uniform lattice, we can compute each sample error as $e_i^o = \|(W_i^o b^o)^T - y_i\|_2^2$ where W_i^o is the i-th row of W^o. The k-th bin ($1 \le k \le N_{bin}$, where N_{bin} is the number of bins) of the normalized error histogram $hist_c$ for channel $c \in \{R, G, B\}$ is computed as:

$$hist_c(k) = \frac{1}{E_c} \cdot \sum_{i\in\Omega_c^k} e_i^o, \tag{6}$$

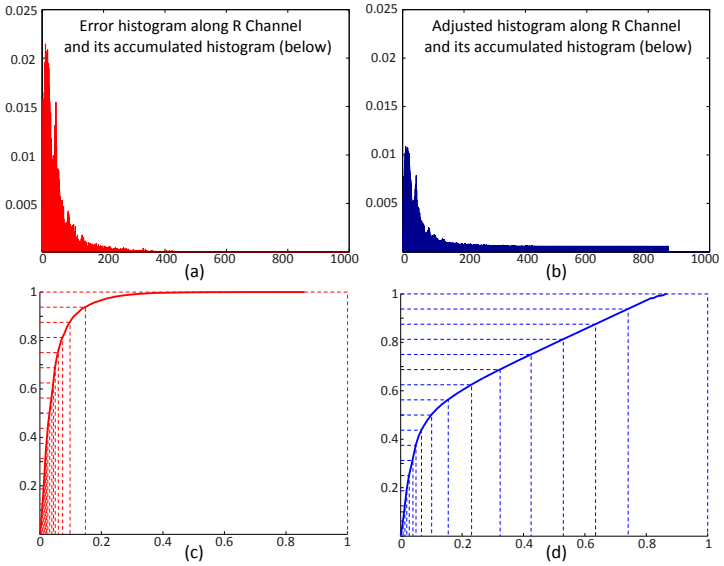

Fig. 4. Illustration of node level transformation function based on error histogram

where E_c is the normalization factor such that $\sum_{k=1}^{N_{bin}} hist_c(k) = 1$ and Ω_c^k contains the indexes of all the sample points that are located in the k-th bin for color channel c. In our experiments, we set $N_{bin} = 1000$.

Our approach uses the accumulative error histogram to construct the node level transformation function. We found, however, that using the $hist_c()$ computed above is problematic as many bins are nearly empty (see Figure 4(a)). This results in flat regions in the accumulative error histogram (see Figure 4(c)). To avoid this problem, we use an adjusted version of the error histogram, $hist_c^*$ which is blended with a constant $(1-\alpha)$ and smoothed by a Gaussian filter (see Eq. 7). Figure 4(b) shows the adjusted histogram.

$$hist_c^*(k;\alpha) = \alpha \cdot Gauss(hist_c)(k) + (1-\alpha)\frac{1}{N_{bin}}. \tag{7}$$

For sake of simplicity, the histogram is treated as a continuous function Θ defined on $[0,1]$, where $\Theta(t;\alpha) = N_{bin} \cdot hist_c^*(k;\alpha)$ when $t \in [\frac{k}{N_{bin}}, \frac{k+1}{N_{bin}})$. We finally define the inverse of node level transformation function as follows:

$$g_c^{-1}(v;\alpha) = \int_0^v \Theta(t;\alpha)dt, \tag{8}$$

where v is a node level. Since Θ incorporates both the error histogram and the uniform term, the derived function g_c^{-1} is able to guide the node level transformation to the erroneous locations while maintaining some nodes in regions of small errors through the control of the uniform term. The monotone increasing property of the accumulated histogram guarantees the existence of a node level

transformation function $g_c(u)$. In our implementation, v can be computed using a binary search over g_c^{-1} for a given u. Figure 4(c) shows the accumulative histogram and the node level transformation functions using the original histogram where α is set to 1. Figure 4(d) shows the accumulative histogram and the node level transformation functions using the adjusted histogram where α is set to 0.5. We can see the function in (d) has a much slower accumulative error histogram function and the lattice levels are not too skewed by the errors.

Nonuniform Lattice Construction. The nonuniform lattice node levels $L_c(\alpha)$ along dimension $c \in \{R, G, B\}$ can be obtained by transforming uniform node levels using $g_c(u)$. The i-th node level $L_c(i; \alpha)$ is calculated as follows:

$$L_c(i;\alpha) = g_c(\frac{i-1}{N_L - 1}; \alpha), i = 1, 2, ..., N_L. \tag{9}$$

Non-uniform lattice $a(\alpha)$ is then constructed as the Cartesian product of $L_R(\alpha)$, $L_G(\alpha)$ and $L_B(\alpha)$, and $b_c(\alpha) = ((\tilde{W}_c^\alpha)^T \tilde{W}_c^\alpha + \lambda K)^{-1} (\tilde{W}_c^\alpha)^T \tilde{y}_c$, where $\tilde{W}_c^\alpha = [W^\alpha; \lambda_S S^\alpha; \lambda_A A_c]$ with W^α representing the weighting matrix of x in $a(\alpha)$ and S^α the new regularization matrix.

While we can set α to a fix value (0.5), we instead let our algorithm choose the best α to blend the nonuniform and uniform arrangements. The value of α can be optimized by solving the following minimization problem using the Trust-Region-Reflective Optimization technique [22, 23]:

$$\hat{\alpha} = \min_{\alpha} \sum_{i=1}^{n} \|(W_i^\alpha b(\alpha))^T - y_i\|_2^2, \tag{10}$$

where W_i^α is the i-th row of W^α. The final lattice is set to be $a(\hat{\alpha})$.

4 Results

In this section, we show several comparisons of our nonuniform lattice regression approach in terms of pixel errors and computation time compared to RBF and uniform lattice regression. All experiments were performed in Matlab and C++ code using a PC with a dual-core 2.3Ghz processor and 3.25GB memory. We used the publicly available calibration data[2] that contains examples of RAW and sRGB images of MacBeth color charts for different camera models under various settings, i.e. exposure, white-balance, and picture styles (when available). This calibration data has 164 color samples per image with up to 10 different exposures and a number of white-balance settings per picture style (e.g. landscape, portrait, standard, vivid, etc). Color samples from the RAW images are first multiplied by their corresponding white-balance matrices. This gives us approximately 10,000 training samples per camera and picture style in the form of $x_i \rightarrow y_i$ as described in Section 2.

[2] Obtained from `http://www.comp.nus.edu.sg/~brown/radiometric_calibration/`

Table 1. Normalized pixel errors and evaluation time comparisons for RBF, uniform lattice regression (LR) and our nonuniform lattice regression (NULR) approach on Nikon examples

NIKON D7000 (size 1632 × 2464)							
Error (pixel)		Average	Maximum	Q3 75%	Q2 50%	Q1 25%	Time (s)
Method	Lattice Size						
Example 1: Picture Style Normal, Exposure Time 1/15 sec, White Balance Tungsten							
RBF	-	0.011790	0.060117	0.006792	0.011092	0.015686	24.2028
NULR	13 × 13 × 13	*0.013215*	*0.061005*	*0.008769*	*0.012401*	*0.017094*	*0.8906*
LR	13 × 13 × 13	0.058985	0.203733	0.032338	0.052467	0.076546	0.8281
NULR	17 × 17 × 17	*0.012502*	*0.068935*	*0.007843*	*0.011765*	*0.016169*	*0.9219*
LR	17 × 17 × 17	0.050673	0.223701	0.025415	0.041130	0.067126	0.8437
NULR	24 × 24 × 24	*0.012172*	*0.064795*	*0.007843*	*0.011765*	*0.016169*	*0.9844*
LR	24 × 24 × 24	0.037690	0.154939	0.017971	0.028818	0.047384	0.8437
Example 2: Picture Style Normal, Exposure Time 1/5 sec, White Balance Fluorescent							
RBF	-	0.009282	0.062868	0.005546	0.008769	0.013006	30.5152
NULR	13 × 13 × 13	*0.012030*	*0.066667*	*0.006792*	*0.011092*	*0.016169*	*0.8906*
LR	13 × 13 × 13	0.041590	0.251470	0.019608	0.033735	0.051729	0.8437
NULR	17 × 17 × 17	*0.011387*	*0.068487*	*0.005546*	*0.009606*	*0.014673*	*0.9062*
LR	17 × 17 × 17	0.031173	0.211110	0.014673	0.023854	0.039411	0.8437
NULR	24 × 24 × 24	*0.010408*	*0.086808*	*0.005546*	*0.008769*	*0.013006*	*0.9375*
LR	24 × 24 × 24	0.022122	0.161405	0.008769	0.016638	0.027730	0.8437
Example 3: Picture Style Landscape, Exposure Time 1/8 sec, White Balance Tungsten							
RBF	-	0.028220	1.006079	0.016169	0.021118	0.027730	7.3593
NULR	13 × 13 × 13	*0.023837*	*0.446319*	*0.014673*	*0.020377*	*0.028006*	*0.8750*
LR	13 × 13 × 13	0.051046	0.460162	0.021479	0.032575	0.077445	0.8281
NULR	17 × 17 × 17	*0.021774*	*0.456353*	*0.013585*	*0.019212*	*0.025110*	*0.8906*
LR	17 × 17 × 17	0.041038	0.455088	0.018394	0.028549	0.052320	0.8437
NULR	24 × 24 × 24	*0.021427*	*0.510241*	*0.013006*	*0.018394*	*0.025110*	*0.9375*
LR	24 × 24 × 24	0.029896	0.488373	0.016169	0.022866	0.036367	0.8437
Example 4: Picture Style Landscape, Exposure Time 1/15 sec, White Balance Sunny							
RBF	-	0.021352	1.000653	0.014139	0.019608	0.025415	6.4687
NULR	13 × 13 × 13	*0.019733*	*0.264437*	*0.012401*	*0.017538*	*0.024174*	*0.8594*
LR	13 × 13 × 13	0.048376	0.224901	0.022866	0.039992	0.070588	0.8437
NULR	17 × 17 × 17	*0.018589*	*0.268334*	*0.011765*	*0.016638*	*0.022528*	*0.8906*
LR	17 × 17 × 17	0.040553	0.215829	0.019212	0.033735	0.057099	0.8281
NULR	24 × 24 × 24	*0.018108*	*0.283766*	*0.011092*	*0.016169*	*0.021479*	*0.9687*
LR	24 × 24 × 24	0.029612	0.232434	0.016169	0.025110	0.039411	0.8437

We compute the initial uniform lattice regression using the method based on Eq. 4. Nonuniform lattice regression is computed using the method described in Section 3. For each method we compute lattices with resolutions of $13 \times 13 \times 13$, $17 \times 17 \times 17$, and $24 \times 24 \times 24$. The one-off computation time in Matlab for the uniform lattice regression at the highest resolution is approximately 7s, while our nonuniform lattice regression required approximately 120s. We use the RBF and tone-curve data provided by the authors [1] on their webpage. To evaluate the RBF, we implemented a cached-optimized method that can reuse colors that have already been computed.

Our results are shown in three tables for the following cameras: Nikon D7000 (Table 1), Canon 1Ds Mark III (Table 2), Sony α200 (Table 3). The different techniques are denoted as nonuniform lattice regression (NULR), uniform lattice regression (LR), and RBF. Each table shows the following normalized pixel error statistics: average error, max error, 25% quartile (Q1) error, 50% quartile (Q2) or median error, and the 75% upper quartile (Q3) error. We also show the running times in seconds (all results computed using C++). The errors are computed on a variety of images with different picture styles, exposure, white-balance settings, and resolutions. We note that Sony α200 has only one picture style on line.

We also show qualitative results in Figure 5, which shows the different pixel errors as a hotmap. Our quantitative and qualitative results demonstrate the our nonuniform lattice regression approach provides performance better than

Table 2. Normalized pixel errors and evaluation time comparisons for RBF, uniform lattice regression (LR) and our nonuniform lattice regression (NULR) approach on Canon examples

CANON 1Ds MARK III (size 1872 × 2808)							
Error (pixel)		Average	Maximum	Q3 75%	Q2 50%	Q1 25%	Time (s)
Method	Lattice Size						
Example 1: Picture Style Standard, Exposure Time 1/15 sec, White Balance Cloudy							
RBF	-	0.009156	0.114332	0.005546	0.008769	0.012401	17.7498
NULR	13 × 13 × 13	*0.009829*	*0.114332*	*0.005546*	*0.008769*	*0.012401*	*1.0625*
LR	13 × 13 × 13	0.045459	0.161595	0.026307	0.039411	0.059344	0.9531
NULR	17 × 17 × 17	*0.009217*	*0.117057*	*0.00554*	*0.008769*	*0.012401*	*1.0937*
LR	17 × 17 × 17	0.028856	0.126467	0.016169	0.026307	0.038822	0.9687
NULR	24 × 24 × 24	*0.009011*	*0.114332*	*0.005546*	*0.008769*	*0.012401*	*1.1719*
LR	24 × 24 × 24	0.018508	0.108819	0.009606	0.016638	0.025110	0.9687
Example 2: Picture Style Standard, Exposure Time 1/10 sec, White Balance Fluorescent							
RBF	-	0.009908	0.103607	0.005546	0.008769	0.013006	20.2185
NULR	13 × 13 × 13	*0.010629*	*0.103607*	*0.006792*	*0.009606*	*0.014139*	*1.0625*
LR	13 × 13 × 13	0.048408	0.199769	0.022866	0.037818	0.059084	0.9531
NULR	17 × 17 × 17	*0.010329*	*0.105007*	*0.006792*	*0.009606*	*0.014139*	*1.0937*
LR	17 × 17 × 17	0.035407	0.174146	0.014673	0.026307	0.042418	0.9687
NULR	24 × 24 × 24	*0.009949*	*0.105007*	*0.005546*	*0.008769*	*0.012401*	*1.1562*
LR	24 × 24 × 24	0.019140	0.110988	0.011092	0.016638	0.024174	0.9844
Example 3: Picture Style Portrait, Exposure Time 1/15 sec, White Balance Tungsten							
RBF	-	0.011571	0.175553	0.006792	0.011092	0.014673	22.5049
NULR	13 × 13 × 13	*0.013200*	*0.170171*	*0.008769*	*0.012401*	*0.017094*	*1.0471*
LR	13 × 13 × 13	0.060657	0.224422	0.028549	0.045901	0.084017	0.9846
NULR	17 × 17 × 17	*0.011961*	*0.178421*	*0.007843*	*0.011765*	*0.016169*	*1.1252*
LR	17 × 17 × 17	0.046053	0.169945	0.023854	0.036367	0.066320	0.9690
NULR	24 × 24 × 24	*0.011522*	*0.170442*	*0.006792*	*0.011092*	*0.014673*	*1.1721*
LR	24 × 24 × 24	0.029428	0.137423	0.017094	0.025110	0.034634	0.9690
Example 4: Picture Style Portrait, Exposure Time 1/5 sec, White Balance Fluorescent							
RBF	-	0.011001	0.215722	0.005546	0.008769	0.014673	29.2521
NULR	13 × 13 × 13	*0.012381*	*0.152084*	*0.006792*	*0.011765*	*0.016638*	*1.0469*
LR	13 × 13 × 13	0.037227	0.327774	0.016169	0.027730	0.042599	0.9688
NULR	17 × 17 × 17	*0.011451*	*0.156224*	*0.005546*	*0.009606*	*0.015686*	*1.0782*
LR	17 × 17 × 17	0.026838	0.271269	0.009606	0.019608	0.034187	0.9688
NULR	24 × 24 × 24	*0.011098*	*0.156961*	*0.005546*	*0.009606*	*0.015686*	*1.1407*
LR	24 × 24 × 24	0.018117	0.168262	0.008769	0.014673	0.023854	0.9844

Table 3. Normalized pixel errors and evaluation time comparisons of RBF, uniform lattice regression (LR) and our nonuniform lattice regression (NULR) approach for Sony examples

SONY α200 (size 1296 × 1936)							
Error (pixel)		Average	Maximum	Q3 75%	Q2 50%	Q1 25%	Time (s)
Method	Lattice Size						
Example 1: Picture Style Standard, Exposure Time 1/15 sec, White Balance Sunny							
RBF	-	0.009144	0.043844	0.005546	0.008769	0.012401	12.9061
NULR	13 × 13 × 13	*0.009999*	*0.069046*	*0.005546*	*0.008769*	*0.012401*	*0.5156*
LR	13 × 13 × 13	0.061349	0.165358	0.041687	0.056693	0.077841	0.4531
NULR	17 × 17 × 17	*0.009889*	*0.080559*	*0.005546*	*0.008769*	*0.012401*	*0.5156*
LR	17 × 17 × 17	0.049572	0.181497	0.029866	0.043844	0.058298	0.4531
NULR	24 × 24 × 24	*0.009811*	*0.127013*	*0.005546*	*0.008769*	*0.012401*	*0.5781*
LR	24 × 24 × 24	0.028771	0.168855	0.016169	0.027730	0.036155	0.4687
Example 2: Picture Style Standard, Exposure Time 1/10 sec, White Balance Fluorescent							
RBF	-	0.009816	0.063233	0.005546	0.008769	0.013006	13.9998
NULR	13 × 13 × 13	*0.010954*	*0.065032*	*0.006792*	*0.009606*	*0.014139*	*0.5156*
LR	13 × 13 × 13	0.053259	0.165498	0.035076	0.050980	0.070806	0.4687
NULR	17 × 17 × 17	*0.010605*	*0.060117*	*0.006792*	*0.009606*	*0.014139*	*0.5312*
LR	17 × 17 × 17	0.044052	0.181497	0.022528	0.036996	0.055459	0.4687
NULR	24 × 24 × 24	*0.010498*	*0.062868*	*0.005546*	*0.009606*	*0.014139*	*0.5625*
LR	24 × 24 × 24	0.028732	0.144673	0.014139	0.024174	0.035727	0.4844

uniform lattice regression and comparable to results obtained with RBF. In terms of time complexity, our approach is an order of magnitude faster than RBF and comparable to using a uniform lattice with a slight overhead of an indirect lookup table to compensate for the nonuniform layout.

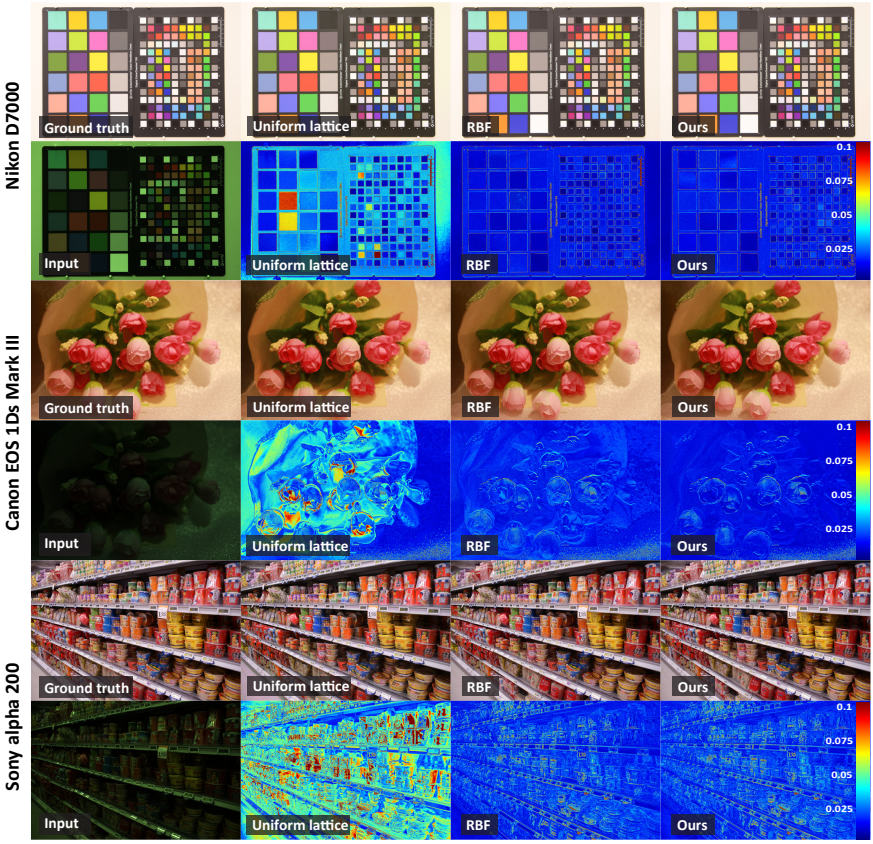

Fig. 5. Several examples from different camera models showing the error maps between our method, RBF, and an LUT based on uniform lattice regression

5 Conclusion and Discussion

We have introduced a novel nonuniform lattice regression approach to compute a sparse LUT for use in modeling the camera imaging pipeline of converting RAW values to their corresponding sRGB output. Our approach is based on a regular lattice design but introduces nonuniform spacing of the lattice layout. Our algorithm adaptively adjusts the lattice grid sampling based on the error histogram to capture the complexity of RAW to sRGB transformation. Our results demonstrate that our nonuniform lattice provides errors comparable to using an RBF, but with computational efficiency similar to a uniform lattice which is an order of magnitude faster than optimized RBF computation. Moreover, the adaptive lattice design allows us to incorporate more steps of the imaging process into the LUT.

Lastly, while our focus was the color mapping pipeline in digital cameras, we believe the ability to simplify complex functions using a nonuniform lattice

can be useful in other applications, such as nonlinear image warping or 3D deformable registration. In additional, the structure of the final nonuniform lattice provides insight into the completely of the underlying warp, that map be useful for function analysis or to help guide better or adaptive calibration methods.

Acknowledgements. We thank the reviewers for their feedback. This work was supported by the NUS Young Investigator Award, R-252-000-379- 101 and the IT Consilience Creative Program of the Ministry of Knowledge Economy, Korea.

References

[1] Kim, S.J., Lin, H.T., Lu, Z., Süsstrunk, S., Lin, S., Brown, M.S.: A new in-camera imaging model for color computer vision and its application. IEEE Transaction on Pattern Analysis and Machine Intelligence (2012)
[2] Grossberg, M.D., Nayar, S.K.: Modeling the space of camera response functions. IEEE Transaction on Pattern Analysis and Machine Intelligence 26(10), 1272–1282 (2004)
[3] Mann, S., Picard, R.W.: On being 'undigital' with digital cameras: Extending dynamic range by combining differently exposed pictures. In: Proc. IS&T 46th Annual Conference, pp. 422–428 (1995)
[4] Debevec, P.E., Malik, J.: Recovering high dynamic range radiance maps from photographs. In: Proc. SIGGRAPH, pp. 369–378 (1997)
[5] Grossberg, M.D., Nayar, S.K.: Determining the camera response from images: What is knowable? IEEE Transaction on Pattern Analysis and Machine Intelligence 25(11), 1455–1467 (2003)
[6] Kim, S.J., Pollefeys, M.: Robust radiometric calibration and vignetting correction. IEEE Transaction on Pattern Analysis and Machine Intelligence 30(4), 562–576 (2008)
[7] Kim, S.J., Frahm, J.M., Pollefeys, M.: Radiometric calibration with illumination change for outdoor scene analysis. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
[8] Manders, C., Aimone, C., Mann, S.: Camera response function recovery from different illuminations of identical subject matter. In: Proc. IEEE International Conference on Image Processing, pp. 24–27 (2004)
[9] Kuthirummal, S., Agarwala, A., Goldman, D.B., Nayar, S.K.: Priors for Large Photo Collections and What They Reveal about Cameras. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 74–87. Springer, Heidelberg (2008)
[10] Lin, S., Gu, J., Yamazaki, S., Shum, H.Y.: Radiometric calibration from a single image. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 938–945 (2004)
[11] Lin, S., Zhang, L.: Determining the radiometric response function from a single grayscale image. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 66–73 (2005)
[12] Chakrabarti, A., Scharstein, D., Zickler, T.: An empirical camera model for internet color vision. In: Proc. British Machine Vision Conference (2009)
[13] Xiong, Y., Saenko, K., Darrell, T., Zickler, T.: From Pixels to Physics: Probabilistic Color De-rendering. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (June 2012)

[14] Urtasun, R., Darrell, T.: Sparse Probabilistic Regression for Activity-Independent Human Pose Inference. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (June 2008)
[15] Garcia, E.K., Gupta, M.R.: Building accurate and smooth ICC profiles by lattice regression. In: Proc. SID/IS&T Color Imaging Conference (November 2009)
[16] Garcia, E.K., Gupta, M.R.: Lattice regression. In: Proc. Advances in Neural Information Processing Systems (NIPS), pp. 1–9 (2009)
[17] Garcia, E.K., Gupta, M.R.: Optimized construction of ICC profiles by lattice regression. In: Proc. SID/IS&T Color Imaging Conference (November 2010)
[18] Chang, J.Z., Allebach, J.P., Bouman, C.A.: Sequential linear interpolation of multidimensional functions. IEEE Transaction on Image Processing, 1231–1245 (September 1997)
[19] Monga, V., Bala, R.: Sort-select-damp: An efficient strategy for color look-up-table lattice design. In: Proc. SID/IS&T Color Imaging Conference (November 2008)
[20] Tastl, I., Recker, J.L., Zhang, Y., Beretta, G.: An efficient high quality color transformation. In: Proc. SID/IS&T Color Imaging Conference (November 2009)
[21] Monga, V., Bala, R.: Algorithms for color look-up table design via joint optimization of node locations and ouptput values. In: Proc. IEEE Int. Conference on Acoustics, Speech and Signal Processing (March 2010)
[22] Coleman, T.F., Li, Y.: On the convergence of reflective newton methods for large-scale nonlinear minimization subject to bounds. Mathematical Programming 67(2), 189–224 (1994)
[23] Coleman, T.F., Li, Y.: An interior, trust region approach for nonlinear minimization subject to bounds. SIAM Journal on Optimization 6, 418–445 (1996)