DYNAMIC OPTIMIZATION ALGORITHM FOR GENERATING INVERSE PRINTER MAP WITH REDUCED MEASUREMENTS

Sohail Dianat*, Lalit. K Mestha #, Athimoottil Mathew*
*Rochester Institute of Technology Rochester NY 14623, #Xerox Corporation, Webster, NY 14580
sadeee@rit.edu, {lkmetha@xeroxlabs.com, avmeee@rit.edu}

ABSTRACT

For a color printer to attain good color rendering quality the image output terminal must be capable of producing the desired tone, i.e., the solidness, of each of primary color separations as requested. Calibration is a major task in providing high quality prints. A model (characterization) of the color printer is a first step for building profiles. Most of the methods used to solve the calibration problem utilize, in one way or another, a printer inverse map. Once the inverse map is constructed, the input image will be processed by the inverse map to obtain the correct *CMY* values that can produce the desired *Lab* values. A major drawback in this type of calibration is the need to measure $N \times N \times N$ color patches each time the printer is calibrated. We propose an alternative approach, where few critical color patches are measured and the rest of the points in the forward printer map are obtained using interpolation.

1. INTRODUCTION

A digital color printer can be seen as a device mapping colors from an input color space into an output color space: e.g., a map from the *CMYK* device – specific color space into the *CIELab* device-independent color space. An approximation to this map can be obtained by performing input-output experiments, by first measuring the printer outputs corresponding to a combination of requested colors at the printer inputs. These requested input colors are the nodes of forward LUT, also known as the characterization LUT. This map is then presented as a LUT that associates points in the printer output color space to points in the printer input color space. Most of the methods used to calibrate printers utilize, in one way or another, a particular form of inverse map.

It is also desirable to have a printer inverse LUT with input nodes regularly spaced on a sequential plane. While measuring the printer forward map, it is possible to structure the input data by selecting equally spaced input grid points. Just by swapping the data of the forward LUT we can generate the inverse, but the data of the input grids for this inverse become unstructured (the output grid of the forward map is unstructured). Another important factor is the accuracy of the inverse map. Accuracy is defined in terms of ΔE between desired input *Lab (in)* and output *Lab (out)*. To achieve an average ΔE less than one for colors inside the printer gamut, the LUT size has to be at least $17 \times 17 \times 17$ for *CMY* to *Lab* printers, since ΔE accuracy is inversely proportional to forward LUT size. To compensate for the printer drift, it is highly desirable to update the inverse printer map on a regular basis. This requires printing $N \times N \times N$ color patches and measuring their *Lab* color values. We propose to down sample the input color space from $N \times N \times N$ to $M \times M \times M$ (M<N) by selecting the so called critical colors such that when we up sample the down sampled LUT to the original size, the average ΔE between the original *Lab* and the up sampled *Lab* values is minimized as shown in Figure.1.

The average ΔE between original *Lab* and *Lab1* is defined as

$$\Delta E = \frac{1}{K} \sum_{i=1}^{K} \left\| Lab(i) - Lab1(i) \right\| \text{, where } K = N \times N \times N \quad (1)$$

These critical colors are selected by minimizing ΔE .Once these critical colors are identified, the updating of the inverse printer map is obtained by first printing and measuring these selected critical patches, then up sample this measured LUT to the full size and then apply an inverse algorithm to obtain the printer inverse map. In this paper, we present a novel patch selection algorithm that is based on dynamic programming.

The rest of the paper is organized as follows. Printer forward model is introduced in Section 2 and details of how it can be generated are discussed. Section 3 presents gamut mapping and inverse printer map. Dynamic optimization (DO) patch selection algorithm is presented in Section 4. In Section 5, simulations of image path with reduced number of measurements are presented.

2. PRINTER FORWARD MODEL

Color device modeling is an essential part of any color management and calibration. There are two types of color device modeling: theoretical and experimental. In theoretical modeling of color devices a small set of color patches are measured and this data is used to extract the parameters of the model. The Neugebauer equations are a classic example of color prediction using a theoretical model [1, 2]. These analytical models are usually not very accurate because they do not adequately capture the nonlinearities of the device due in large part to external variables such as temperature, humidity, and the parameters of the print media. Another common modeling approach is the use of a LUT. A color printer can be viewed as a device that maps colors from an input color space to an output color space. The printer can be modeled by a LUT containing the points in the input color space and their corresponding points in the output color space. For a three-color (*CMY*) printer, the input color space is device dependent *CMY* and the output is device- independent *CIELab* color space. The printer forward map LUT is a transformation from *CMY* to *Lab*, denoted by P , and is normally measured by printing $N \times N \times N$ uniformly spaced *CMY* color patches and measuring their corresponding *Lab* values. The *CMY* patches are first transformed to *CMYK* by an appropriate grey color replacement and under color removal (GCR/UCR) transformation

that consists of black addition and *CMY* subtraction. The *CMYK* patch is then printed by the print engine. The forward printer model LUT is denoted by P and is shown in Figure 2. The GCR/UCR used is a simple algorithm. We first subtract $0.5x^2$ from three-components of *CMY* and add black colorant $K = x^2$, where $x = \min(C, M, Y)$.

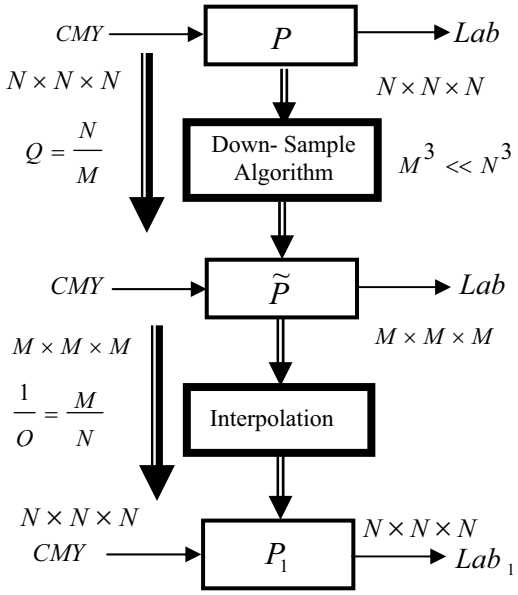

Figure 1: Down Sampling and Up Sampling of *P*

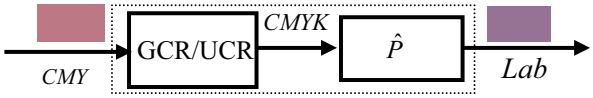

Figure 2: Forward Printer Model

3. INVERSE PRINTER MODEL

The forward printer model P is inverted to obtain a device correction function, which maps each device-independent color *Lab* to the device dependent values *CMY*. This is defined mathematically as

$$P^{-1} : Lab \rightarrow CMY \quad (2)$$

where input *Lab* points are placed on a uniformly sampled 3-d grid of size $N \times N \times N$ having a dynamic range of $0 \le L \le 100$, $-127 \le a, b \le 128$. This inverse map is a critical part of the calibration algorithm. Different algorithms can be used to compute the printer inverse map. They include Shepard [3], Moving Matrix [4], Iteratively Clustered Interpolation (ICI) [5], and Conjugate Gradient algorithm. Out of gamut *Lab* values are first mapped to the nearest gamut boundary points using an appropriate gamut-mapping algorithm, which preserve brightness and hue. If the original color in the *CIELab* color space is denoted by $[L\ a\ b]$ and it's corresponding mapped by $[L'\ a'\ b']$, then

$$L' = L, \quad \tan^{-1}\left(\frac{b'}{a'}\right) = \tan^{-1}\left(\frac{b}{a}\right) \quad (3)$$

The mapping algorithm is performed in two steps. In the first step, we determine whether a given input color is inside or outside the printer gamut. This is achieved by: 1) Transforming the color gamut into spherical coordinates where the center of the sphere is at the centroid of the gamut, approximately equal to *Lab= [50 0 0]*. In these new color coordinates, each color is specified by its spherical coordinates r, α, and θ where, r is the distance from the gamut centroid to the given color point, α is the hue angle with the dynamic range from 0 to 2π and θ is the angle in the constant α plane from 0 to π. 2) Finding the intersection of the gamut with a straight line connecting the input color point to the centroid of the gamut. This is achieved by searching the gamut boundary points with coordinates that fall into the range $\alpha \pm \Delta\alpha$ and $\theta \pm \Delta\theta$. The average of these boundary points is denoted by $[r_{ave}\ \alpha_{ave}\ \theta_{ave}]$. The desired color point is inside the gamut if $r_0 < r_{ave}$, otherwise it is outside. Once we determine that the color is outside the printer gamut, we map it to the nearest point on the gamut boundary that best satisfies the conditions given by (2).The mapped *Lab* values are then inverted using an inversion algorithm. This will produce a LUT from *Lab* to *CMY*, where the input *Lab* is on a uniform grid. The ICI algorithm used in this paper is a gradient-based iterative algorithm, which uses a clustering approach for initialization of the algorithm [3]. With *J(k)* being the Jacobian of transformation *P,* the updating equation

$$CMY(k+1) = CMY(k) - \mu J(k)\{P[CMY(k)] - Lab\} \quad (4)$$

4. PATCH SELECTION ALGORITHM

The 3-d transfer function P of a typical color printer is nonlinear and can even be time varying, due to the interaction of the printer colorants and paper substrate, changes in temperature, and humidity. The printer static calibration can be achieved by designing a look-up table to approximate the inverse printer map (inverse of $\tilde{P}$). In an open-loop system calibration, the input image is passed through calibration transformation (inverse of $\tilde{P}$) before being fed to the printer for reproduction. This scheme will achieve a ΔE close to zero between an input *Lab(in)* patch and measured (output) *Lab(out)* patch on the paper, provided that the printer does not drift and the inverse printer LUT is a high-resolution LUT such that the interpolation error can be neglected. None of these ideal conditions exist in the real world. For these reasons, we need to design a dynamic controller to operate on a finite number of color measurements and a fast interpolation algorithm for use with uncontrolled colors. A practical approach is to select a finite number of patches in color space and control these points. If these selected points are chosen properly, the whole color space can be reconstructed with a small MSE by using an appropriate mapping algorithm. There are different patch selection algorithms such as Sequential Linear Interpolation [6], and Piece-wise Linear Homeomorphisms [7]. The patch selection algorithm selects a finite number of points by minimizing the MS error (ΔE) between the actual printer output and the up-sampled printer output constructed using finite number of points as defined by

$$\Delta E = \left\| Lab(out1) - Lab(out2) \right\| \quad (5)$$

$$Lab(out1) = P(CMY) \quad (6) \quad Lab(out2) = \hat{P}(CMY) \quad (7)$$

The LUT ($\hat{P}$) is obtained by up-sampling the smaller LUT containing the finite number of points (critical patches) as selected by an appropriate patch selection algorithm. This is shown in figure 3. Once the up-sampled LUT ($\hat{P}$) is constructed, its inverse

LUT (inverse of $\hat{P}$) is computed using Iterative ICI algorithm.

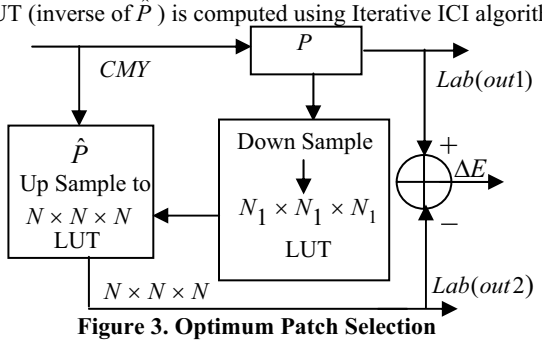

Figure 3. Optimum Patch Selection

Patch selection DO algorithm is based on dynamic programming, which is a multistage decision process. Details of DO algorithm will be presented in the journal version of this paper. Here we present a summary of the algorithm. We first discuss the one-dimensional case and then extend it to two and three dimensions.

4.1 One-Dimensional DO Algorithm

Consider a 1-d discrete function $f(x)$, where x takes M discrete values $x_1, x_2, \cdots, x_M$. We would like to choose $N < M$ points $\hat{x} = [\hat{x}_1 \quad \hat{x}_2 \quad \cdots \quad \hat{x}_N]$ such that $\hat{x}_1 = x_1, \hat{x}_N = x_M$, and $\hat{x} \subset x$, while minimizing the mean square error resulting from piecewise linear approximation defined by $\hat{f}(\hat{x})$ and $f(x)$ over x. The mean square error MSE is given by

$$E = \frac{1}{M}\sum_{i=1}^{M}\left|f(x_i) - L_k(x_i)\right|^2 \quad (8)$$

where $L_k(x)$ is the straight-line segment joining $f(\hat{x}_k)$ to $f(\hat{x}_{k+1})$ and is given by

$$L_k(x) = \frac{f(\hat{x}_{k+1}) - f(\hat{x}_k)}{\hat{x}_{k+1} - \hat{x}_k}(x - \hat{x}_k) + f(\hat{x}_k), \quad (9)$$

for $\hat{x}_k \le x \le \hat{x}_{k+1}$. Since we always need the first and the last points for interpolation, the number of grid points has to be greater than or equal to three $N \ge 3$. Now assume that *N=3*, this means that we need to find one grid point $\hat{x}_2$ between x_1 and x_M such that the total MSE given by (8) is minimized. Let the solution be x_j, where $1 < j < N$. The MMSE is given by

$$E^* = \frac{1}{M}\sum_{i=1}^{j}\left|f(x_i) - L_1(x_i)\right|^2 + \frac{1}{M}\sum_{i=j+1}^{M}\left|f(x_i) - L_2(x_i)\right|^2 \quad (10)$$

where $L_1(x)$ and $L_2(x)$ are

$$L_1(x) = \frac{f(x_j) - f(x_1)}{x_j - x_1}(x - x_1) + f(x_1) \quad (11)$$

$$L_2(x) = \frac{f(x_M) - f(x_j)}{x_M - x_j}(x - x_j) + f(x_j) \quad (12)$$

Index j and E^* is found by the following optimization problem

$$j = \underset{k}{\operatorname{argmin}}\,(E(k)) = \underset{k}{\operatorname{argmin}}\,\frac{1}{M}\sum_{i=1}^{k}\left|f(x_i) - L_1(x_i)\right|^2 + \frac{1}{M}\sum_{i=k+1}^{M}\left|f(x_i) - L_2(x_i)\right|^2 \quad (13)$$
$$E^* = E(j)$$

This means that if we start from x_1 and wish to locate one grid point between x_1 and x_M to approximate $f(x)$, that point will be x_j and the corresponding MMSE will be E^*. We define new index $j_1 = j$ and error $E_1 = E^*$ and assign these two numbers to x_1 and write it as $\{j_1 \; E_1\}$. We repeat this process for x_2 through x_{N-2} and form the following array of numbers as shown in column 1 of Table 1. We call this the 1-d single stage grid allocation algorithm. In this array, E_i is the MSE between the original function and linearly interpolated function using grid points $[x_i \; x_j \; x_N]$ over the closed interval of $[x_i \; x_N]$. These indices and their corresponding MSE are required for the two stage optimal search. In the two stage search, we try to locate two optimal grid points between x_i and x_N such that the total MSE is minimized. Using dynamic programming we need to minimize

$$k_i = \arg\min_m [E_{im} + E_m], \quad m \ge i+1$$
$$EE_i = E_{ik_i} + E_{k_i}, \; l_i = j_{k_i} \quad (14)$$

where E_m is the MMSE corresponding to *j=1*, and E_{im} is the MSE between the original function and linearly interpolated function using grid points $[x_i, x_m]$ over the interval of $[x_i, \cdots, x_m]$. Then the optimum 2-point grid including boundary points will be $[x_1 \; x_k \; x_l \; x_N]$ as shown in column 2 of Table 1. We call this the 1-d two stage grid allocation algorithm. Using the same approach, we can now build an *n*-stage optimization process for the 1-d case with the optimal N grid points $[x_1 \; x_{m_1} \; x_{n_1} \cdots x_{N_1}]$.

Point	*J=1*	*J=2*	*J=…*	*J=N-2*
x_1	$[j_1, E_1]$	$[k_1, l_1, EE_1]$	…	$[m_1, n_1, \ldots, E_{Total}]$
x_2	$[j_2, E_2]$	$[k_2, l_2, EE_2]$	…	-
x_3	$[j_3, E_3]$	$[k_3, l_1, EE_3]$	…	-
⋮	⋮	⋮	⋮	⋮
x_{N-4}	$[j_{N-4}, E_{N-4}]$	$[k_{N-4}, l_{N-4}, EE_{N-4}]$	…	…
x_{N-3}	$[j_{N-3}, E_{N-3}]$	[N-2, N-1, 0]	-	-
x_{N-2}	[N-1, 0]	-	-	-
x_{N-1}	-	-	-	-
x_N	-	-	-	-

Table 1. 1-d DO Algorithm

4.2 Two-Dimensional DO Algorithm

Consider the 2-d discrete function $f(X)$, where $X = [x, y]$ takes $M \times M$ grid points uniformly spaced in $x - y$ plane as shown in Figure 4. We would like to choose $N \times N$ grid points out of $M \times M$ available data points such that the mean square interpolation error between the original function and the approximation obtained by up sampling the $N \times N$ LUT is

minimized. Since the boundary points should be included for interpolation, the degrees of freedom are less than $N \times N$. For example, if *A* is an internal optimum point belonging, then the four boundary points B, *C, D* and *E* (as shown in Figure 4) should be also included in this set. The 2-d DO algorithm is very similar to the 1-d case. In the single stage 2-d algorithm, assume we start with point $[x_i, y_j]$ and try to optimally select one internal grid point in the set extending from this point to the boundaries of the function *f(X)*. The solution is obtained by direct optimal search and the solution is denoted by the two indices k_i and l_j with its associated four boundary points and MMSE $J_1(i,j)=E_{i,j}$. This process is repeated for each point in the set associated with the domain of the underlying function. With three indices given by

$$[k_i, l_j, E_{i,j}] \quad i, j = 1, 2, \cdots, M \quad (15)$$

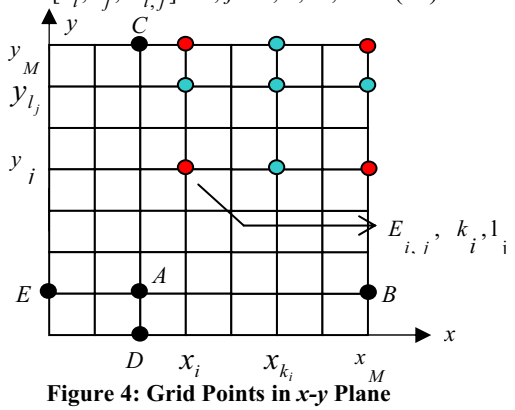

Figure 4: Grid Points in *x-y* Plane

In the second stage we will try to select two internal points in the same set. If the indices corresponding to the first optimal point are n and m, then the total MSE is

$$E = E_{(i,n),(j,m)} + E_{n,m} \quad (16)$$

where $E_{(i,n),(j,m)}$ is the MSE resulting from interpolating all the grid points between indices i to n and *j* to *m* using all necessary boundary points. We now minimize *E* in (14) with respect to *i* and *j*. Let the solution be $i = n_i$, $j = m_j$. Then the indices n_i, m_j, n, m correspond to the two optimum internal points (total of 14 points including boundary points) with MSE given by

$$E_2(i,j) = E_{(n_i,n),(m_j,m)} + E_{n,m} = E_{(n_i,n),(m_j,m)} + E_1(n,m) \quad (17$$

This process is continued until we select *N* grid points.

4.3 Three-Dimensional DO Algorithm

Consider a 3-d discrete function $f(X)$ represented by a set of pairs $(X, f(X))$ where X is 3-d vector. These points are uniformly spaced in the support space of $f(X)$ forming an $N \times N \times N$ LUT. The goal is to down sample this 3-d LUT to a smaller LUT of size $M \times M \times M$ while minimizing the MSE between the original and the up sampled LUT. The 3-d DO algorithm, which is an extension of the 2-d algorithm, can be used for optimally selecting these grid points. Similar to 2-d cases, it is a multistage decision process described by: a) In the single stage decision process, assume that, we start from point $\{i, j, k\}$ and form a cube extending this point to the boundaries of the underlying function. The goal is to find one point inside the cube with its associated boundary points such that the MSE between the interpolated and original grid point values is minimized. Let the optimal indices be l_i, m_j, n_k with MSE of E_{ijk}. b) In the second stage, we need to find two grid points within the same cube in order to minimize the MSE as in stage one. If the solution for the first point has indices l_{1i}, m_{1j}, n_{1k} then the total MSE is given by

$$E_2(l_{1i}\ m_{1j}\ n_{1k}) = E(i,j,k) + E_{l_{1i} m_{ij} n_{1k}} \quad (18)$$

where $E(i,j,k)$ is the interpolation error between the original and the interpolated function with the cube extending from grid point $\{i, j, k\}$ to grid point l_{1i}, m_{1j}, n_{1k}. Now, we minimize E_2 with respect to the indices l_{1i}, m_{1j}, n_{1k}. Note that the solution to the first stage is used for finding the optimal solution in the second stage similar to 1-d and 2-d algorithms. c) We continue this process until we find the solution to the *N*-stage problem. Note that while finding the solution to the *N*-stage grid allocation problem, we obtain the solution to all stages up to and including *N*.

5. SIMULATION RESULTS

The results of the simulation for 512×512 Lena image using Xerox Phase 770 digital printer is shown in Table 2. Details of the experiment and conclusions will be presented at the conference.

# of Patches	Mean ΔE	Min ΔE	Max ΔE	Mean +2std	Median ΔE	95% tile
3^3	3.11	0.27	8.84	5.03	2.98	4.69
4^3	2.09	0.04	8.94	4.14	1.71	3.93
6^3	1.18	0.02	7.05	2.95	0.74	2.79
13^3	1.03	0.00	8.80	3.24	0.43	3.12

Table 2: Profiling Using DO Algorithm

6. REFERENCES

[1] R. Balasubramanian, "The use of spectral regression in modeling halftone color printers", IST/OSA Annual Conference, Optic/Imaging in the Inform Age Rochester, NY, October 1996, pp. 372-375.

[2] M. Xia, E. Saber, G. Sharma, and M. Tekalp, "End-to-end color printer calibration by total least squares regression", IEEE Trans on Image Processing (1999), no. 5, pp. 700-716.

[3] Donald Shepard, "A two-dimensional interpolation function for irregularly- spaced data", Proc. ACM National Conf, pp. 517 –524, 1968.

[4] R. Balasubramanian, and M. S. Maltz, "Refinement of printer transformation using weighted regression," Proc of SPIE, 96.

[5] D. Viassolo, and L. K. Mestha "A practical algorithm for the inversion of an experimental input-output color map for color correction," *Journal of Optical Engr.* vol. 42, no. 3, March 03

[6] J.Z. Chang, "Sequential Linear Interpolation of Multidimensional Functions," IEEE Trans on Image Processing, vol. 6, pp. 1231-1245, September 1997.

[7] R.E. Groff, D.E.Koditschek, and P.P. Khargonekar, "Piecewise linear homeomorphisms: The scalar case", IEEE Intl Conf on Neural Networks, vol3, pp. 259-264, July 2000.