7. International Immersive Projection Technologies Workshop
9. Eurographics Workshop on Virtual Environments (2003)
J. Deisinger, A. Kunz (Editors)

Color Gamut Matching for Tiled Display Walls

Grant Wallace, Han Chen and Kai Li

Department of Computer Science, Princeton University, Princeton, NJ 08544.
{gwallace,chenhan,li}@cs.princeton.edu

Abstract

This paper presents a non-parametric full-gamut color matching algorithm. Color matching is important for the seamless appearance of tiled displays. In particular we address the case where the tiled display is composed of different types of projectors or DLP projectors with white enhancement. White enhancement produces a non-additive color space that is difficult to model. We perform our calibration using an inexpensive colorimeter as opposed to a highly accurate spectroradiometer. Our results show that we can achieve good color balance with 1.47% variance between projectors. We present a method for applying this color gamut mapping in real-time on the newest commodity graphics cards.

Categories and Subject Descriptors (according to ACM CCS): I.3.2 [Graphics Systems]: Distributed/Network Graphics I.3.3 [Computer Graphics]: Picture/Image Generation—Display Algorithms, Viewing Algorithms I.3.7 [Three-Dimensional Graphics and Realism]: Color, Shading, Shadowing, and Texture

1. Introduction

Large format high-resolution display devices are becoming increasingly important for scientific visualization, industrial design and entertainment applications. A popular approach to building such displays is the *projector array*[3], where several commercially available projectors are tiled to create a seamless, high-resolution display surface, as shown in Figure 1.

Tiled projector arrays require precise calibration in order to appear seamless. There are three main aspects of projector calibration: geometric alignment, luminance balancing, and color matching. Geometric alignment is needed to remove the discontinuity caused by mis-aligned projectors in the overlapped regions. Several vision based software solution[1, 2, 7, 10] were proposed to address this problem. However, even with sub-pixel accurate geometric alignment, photometric imbalance within and among the projectors can still cause obvious and severe visual artifacts, thus reducing the overall effectiveness of such a display. Majumder *et al.* proposed the use of *Luminance Attenuation Map* (LAM) to equalize the luminance output across the display wall[6]. Majumder *et al.*[5] presented a generalized description of the problem and proposed a method to partially address the issue of color matching through an independent intensity matching on red, green and blue channels of all projectors. Stone[9] presents an algorithm for finding the standard gamut of LCD projectors and gives a characterization of the problems DLP projectors present in color balancing. Stone also proposes *Independent Channel Balancing* (ICB). As noted in Stone[9], channel balancing assumes chromaticity constancy and an

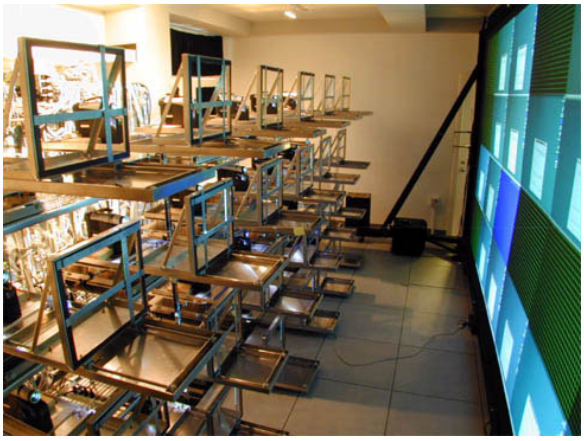


Figure 1: *Princeton Scalable Display Wall. There are 24 DLP projectors giving an overall resolution of* 6000×3000.

additive gamut. Thus, they only work for a homogeneous array of LCD projectors from the same manufacturer.

However, for a display wall containing DLP projectors or LCD projectors from different vendors, color matching becomes critical. First of all, the chromaticity values of the RGB primaries of projectors from different vendors are typically different. Second, commodity single-chip DLP projectors use a clear channel on the color wheel to boost the light output for bright colors[4, 8]. This results in a non-additive gamut, as shown in Figure 2, which is no longer a parallelepiped in the CIE XYZ space, and may even be a concave polyhedron. As such a matrix transformation will not work and a generalized mapping is needed.

Another issue in color matching is the gamma correction of the projectors. Previous luminance balancing and color matching methods rely on the linearity of the color transfer function of projectors. This means "gamma correcting" each projector to have a 1.0 gamma. Although this is mathematically simpler to deal with, it is not necessarily visually appealing, as the human visual system is more adapted to a gamma of 1.8 to 2.2. It also causes lost precision for brighter colors because of limited bits for the *Look-Up Table* (LUT).

This paper presents a practical solution for color matching display walls made of DLP projectors or mixed vendor/technology projectors. We use an inexpensive colorimeter to measure the color gamut of each individual projector. We then use a non-parametric model to find a color mapping for each projector to achieve a common color gamut. Finally, we propose implementation methods for applying the color map along with previously proposed geometric alignment and luminance balancing in real-time with graphics hardware. The non-parametric model enables us to color match projectors with different primaries and/or non-additive gamuts. When homogeneous LCD projector arrays are used, a linear model can also be extracted from the data, such as that suggested in[9]. The non-parametric color matching method is also able to preserve the gamma value.

The remainder of the paper is organized as follows. Section 2 studies the color characteristics of DLP projectors and discusses the challenges in color matching them. Section 3 presents our non-parametric color matching system for display walls. Some experimental results are shown in section 4. Section 5 concludes the paper, and suggests future work.

2. Color Matching DLP Projectors

In this section, we first define the general color matching process. We then discuss the challenges involved in dealing with commodity DLP projectors.

2.1. Generalized Color Matching Process

The color reproduction process of a display system can be described as follows. First, an RGB triple (r,g,b) in the graphics frame buffer is converted to either an analog voltage or digital bits and sent to the projector. The projector then combines lights of three primary colors proportionally to form the desired color. Given the spectrum of the output light, we can calculate the tristimulus values (X,Y,Z), which reflect the response of a typical human visual system. The entire process can be characterized as a *Color Transfer Function* $F\colon R^3 \rightarrow R^3$, $(X,Y,Z) = F(r,g,b)$. F maps a value (color) from RGB space to the CIE XYZ space.

The gamut of a display device is the set of all reproducible colors. Commodity graphics hardware typically have an 8-bit depth in each of the RGB channels, $r,g,b \in [0,1,\ldots,255]$. Hence, the gamut of a color transfer function F can be formally defined as

$$G(F) = \{F(r,g,b) | r,g,b \in [0,255]\}.$$

We call a gamut an *additive gamut*, if it satisfies that

$$F(r,g,b) = F(r,0,0) + F(0,g,0) + F(0,0,b).$$

That is, the three RGB channels are independent of each other.

Further, if the device gamma is set to 1.0, the transfer function becomes linear. It can then be expressed as a matrix transformation.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = [f_{ij}] \begin{bmatrix} r \\ g \\ b \end{bmatrix} + \begin{bmatrix} r_0 \\ g_0 \\ b_0 \end{bmatrix}$$

where $[r_0, g_0, b_0]^T$ is the constant *black offset*. When a device does not have an additive gamut or the gamma is not 1.0, there is no such matrix $[f_{ij}]$ that satisfies the mapping.

For a tiled display to look seamless, all projectors in the system must reproduce colors in the same way, that is, they should all share a common color transfer function. However, this is usually not true in practice. Thus, there exists the need to color match the projectors.

Without access to the graphics card and projector hardware, color matching can be achieved through the use of a *color map* $M\colon [0,1,\ldots,255]^3 \rightarrow [0,1,\ldots,255]^3$, $(r',g',b') = M(r,g,b)$. The color map is applied to pixels before they are sent to the display. The equivalent color transfer function of the system can now be expressed as $F \circ M$. Given n projectors in a tiled display, each with a color transfer function of F_i, the color matching problem can be formally stated as: find M_i for $i = 1,\ldots,n$, such that $F_i \circ M_i = F_j \circ M_j$, $\forall i,j \in \{1,\ldots,n\}$.

2.2. Characteristics of DLP Projectors

LCD projectors usually have an additive gamut. Therefore, the color matching can be easily achieved through an 3×3 matrix multiplication in the RGB space, provided that the gamma is "corrected" to 1.0.

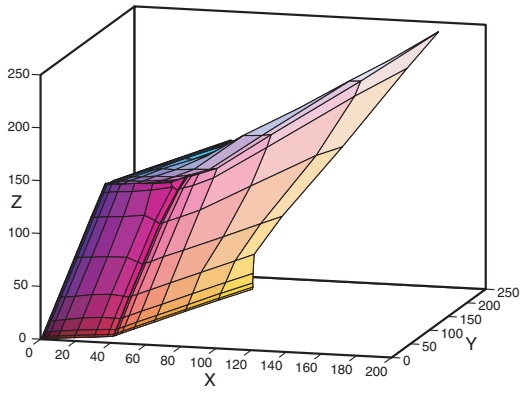


Figure 2: *Color Gamut of a Typical DLP Projector*

DLP projectors can be more difficult to color match than LCD projectors. Commodity single-chip DLP projectors use a spinning color wheel with primary color filters to create color channels in a time sharing fashion. They commonly use a method called "white enhancement" to increase the contrast ratio of the projector—in addition to the Red, Green, and Blue filters, a fourth White (or Clear) filter is added to the color wheel, which passes the full spectrum of the projector bulb. This is similar to the CMYK color printing process, where Cyan, Magenta, Yellow, and Black inks are used. The DLP projector chip controls how much white to add based on a function of the input RGB pixel value. As white is added, output RGB values are reduced correspondingly. Current DLP chips use a step function, adding white in 4 discrete increments.[4]

One result of using white enhancement is that DLP projectors will exhibit different white points even after indepedent channel balancing is performed. This is due to the different spectral outputs of the bulbs, which is the result of either manufacturing tolerances or bulb decay over its lifetime.

The color gamut of a typical DLP projector is shown in Figure 2. As can be seen from the figure, the gamut does not form a parallelepiped in XYZ space and so is not an additive gamut. The gamut becomes stretched towards the white point due to the white enhancement. In order to map a gamut like this to XYZ space a non-parametric mapping is needed.

3. Non-parametric Color Matching System

As described in the previous section, commodity DLP projectors typically have non-additive color gamuts. Even though it is possible to model a DLP projector gamut with a piece-wise linear model, the underlying parameters are device dependent and proprietary to the manufacturer. Therefore, we treat the color transfer function $(X,Y,Z) = F(r,g,b)$ of the projectors as a black box. For simplicity, we assume monotonicity of F for each variable.

3.1. Measuring Color Transfer Function

With a non-parametric model, one has to measure the XYZ value for each of the 2^{24} possible input RGB value combinations. However, this is infeasible to implement. Practically, we sample F at a lower spatial frequency and use interpolation to fill in the values in between.

Because of the gamma curve, F changes slowly at the low end of RGB, but increases faster as RGB values grow. To accommodate this, we use a non-uniform sampling grid, with denser sampling intervals at the high end of input RGB values.

We use a colorimeter to measure the chromaticity value of an input RGB value. The colorimeter returns the x, y, z, and Y value of the input color, and we calculate the X, Y, Z value as follows

$$\begin{aligned} X &= xY/y \\ Y &= Y \\ Z &= zY/y \end{aligned}$$

3.2. Standard Color Transfer Function

Assuming monotonicity of a color transfer function F, which should be true when the projector's brightness and contrast settings are not saturated, its gamut $G(F)$ is the volume in XYZ space bounded by the following six surfaces

$$\begin{aligned} S_1 &= \{F(0,g,b)|g,b \in [0,255]\} \\ S_2 &= \{F(255,g,b)|g,b \in [0,255]\} \\ S_3 &= \{F(r,0,b)|r,b \in [0,255]\} \\ S_4 &= \{F(r,255,b)|r,b \in [0,255]\} \\ S_5 &= \{F(r,g,0)|r,g \in [0,255]\} \\ S_6 &= \{F(r,g,255)|r,g \in [0,255]\} \end{aligned}$$

We use a triangle mesh generated from the sampled F data to form a polyhedral representation of G.

Let $G_i = G(F_i)$ be the color gamut of the ith projector. The common color gamut G_c that can be reproduced by all projectors is therefore the intersection of all G_i:

$$G_c = G_1 \cap G_2 \cap \cdots \cap G_n$$

By applying the polyhedron intersection algorithm, we obtain a polyhedron representing G_c. Note that, because G_i can be concave, as in the case of DLP projectors, the intersection operation might produce a set of disjoint polyhedrons. In this case, we simply use the polyhedron with the largest volume as G_c, and discard the rest. This is dictated by the implied continuity requirement of F_c.

Once we have the common color gamut G_c, we can find a standard color transfer function F_s. The goal is to maximize the volume of $G(F_s)$, with the constraint that $G(F_s) \subseteq G_c$.

To describe the algorithm of finding F_s, we first define a projective transform $H: R^3 \rightarrow R^3$.

$$x' = \frac{h_{11}x + h_{12}y + h_{13}z + h_{14}}{h_{41}x + h_{42}y + h_{43}z + 1}$$
$$y' = \frac{h_{21}x + h_{22}y + h_{23}z + h_{24}}{h_{41}x + h_{42}y + h_{43}z + 1}$$
$$z' = \frac{h_{31}x + h_{32}y + h_{33}z + h_{34}}{h_{41}x + h_{42}y + h_{43}z + 1}$$

We call two color transfer functions *projectively related*, if there exists a projective transform H, such that

$$F_1 = H \circ F_2$$

The algorithm is then described as below

1 Pick one of the color transfer functions, say F_1.
2 For each F_i, find an H_i such that the L2 distance of F_1 and $H_i \circ F_i$ is minimized.
3 Let $\bar{F} = \frac{1}{n}\sum_{i=1}^{n}(H_i \circ F_i)$
4 Maximize the volume of $G(H_s \circ \bar{F})$, with respect to H_s and with the constraint that $G(H_s \circ \bar{F}) \subseteq G_c$
5 The standard color transfer function is $F_s = H_{s,max} \circ \bar{F}$

To put the algorithm in plain English, we first obtain a starting color transfer function $\bar{F}$ by averaging F_i normalized to the shape of F_1, and then find the standard (common) color transfer function by warping $\bar{F}$, and maximizing its volume with the constraint that it has to be contained by the common color gamut G_c.

Starting from the average shape of all color gamuts allows us to preserve all the properties of the original color transfer function, such as its gamma.

3.3. Generating Color Maps

In order to emulate the standard color transfer function $\bar{F}$ on each of the projectors, a color map M is applied on the imageries before they are displayed, as discussed in Section 2.1. For a color map to be feasible, it has to satisfy the following condition

$$\forall (r,g,b) \in [0,255]^3, \quad M(r,g,b) \in [0,255]^3$$

That is, the color map never produces out-of-gamut colors. The goal of the color map is such that $F_i \circ M_i = F_s$.

Therefore,

$$M_i = F_i^{-1} \circ F_s$$

Note that:

$$\forall (r,g,b) \in [0,255]^3, \quad F_s(r,g,b) \in G(F_s) \subseteq G_c \subseteq G_i$$

Thus, $M_i(r,g,b) \in F_i^{-1}(F_s(r,g,b)) \in [0,255]^3$.

That is, M_i is indeed feasible with the definition of F_s.

Because we can only sample F_i at some discrete points, interpolation is needed when a value is not directly sampled in F_i. The final color map is a discretized version of M_i defined on $[0,1,\ldots,255]^3$ to itself.

3.4. Applying a Color Map with Graphics Hardware

Applying the color map in CPU is a costly operation, which precludes the possibility of real-time software color matching. The advance in graphics hardware, especially the programmable Pixel Shader in DirectX or Texture Shader in OpenGL, has enabled us to apply the color mapping in the graphics card.

To achieve this, we load the discretized color map M as a volume texture. For each pixel, we treat its (r,g,b) color value as a volume texture coordinate (u,v,w), and sample M to find out the mapped color. This operation can be implemented with the `texreg2rgb` instruction available in the Microsoft DirectX Pixel Shader Language version 1.2 and 1.3.

As mentioned in Section 1, color matching is one aspect of the overall projector calibration process. To combine the geometric alignment, luminance balancing and color matching together, we propose the following rendering architecture.

1 3D scenes are rendered to a texture. In the case of 2D applications, images or video frames are loaded into a texture buffer. This is the first texture stage.
2 The discretized color map is loaded into a volume texture, and used as the second texture stage.
3 The luminance map is loaded into texture as the third texture stage.
4 Set up the first texture combiner to copy the first texture in decal mode.
5 Set up the second texture combiner to sample the volume texture using the output of the first stage as texture coordinates.
6 Set up the third texture combiner to multiply the output of the second stage with the third texture.
7 Set up the view and projection matrices to represent the geometric pre-warping.
8 Draw a rectangle.

The corresponding pixel shader is shown in Figure 3.

Our tests, presented in Section 4.2.3, indicate that the latest graphics cards, such as the NVIDIA GeForce4 Ti4600, can support such operations for full frame images at real time. Future versions of the Pixel Shader language will allow more instructions, flow control and floating point precision color. With these additions, we expect that higher quality calibration can be achieved.

```
ps.1.2
// t0 is the rendered texture
tex t0
// t1 is the color map
texreg2rgb t1, t0
// t2 is the luminance map
tex t2
mov r1, t1
mul r0, t2, r1
```

Figure 3: *PixelShader code for applying the color map, luminance map, and geometric warping*

4. Implementation and Results

In this section we compare our *Full-Gamut Color Matching* algorithm (referred to as FGCM afterwards) with the *Independent Channel Balancing* algorithm (refered to as ICB afterwards) for two test cases. In the first test case we use a uniform array of 4 DLP projectors, and in the second case we use a mixed array consisting of one DLP projector and one LCD projector. For these tests we implemented the FGCM algorithm in Matlab, and for comparison we implemented an ICB algorithm, such as that described in[5, 9]. Section 4.1 details the measurement method and metrics we use to measure the accuracy of these algorithms and Section 4.2 gives experimental results.

4.1. Experimental Setup

Our color matching experiments consist of three steps. First we measure a subsample of the color gamut of each projector. Next we compute a color map using either FGCM or ICB. Finally we apply the appropriate map to each projector and re-measure the color gamuts. We then compute an error metric to determine the accuracy of the transformations.

In the first step of our experiment we measure the color gamut of the projectors. To do this we use an inexpensive colorimeter, the Sequel Imaging Chroma IV, which reads the color response in CIE XYZ space. We subsample the projector RGB domain with a 32 increment for RGB values less than 128, and a 16 increment for values greater than 128. This gives us 13 points in each of the R, G, and B channels, that is, 0, 32, 64, 96, 128, 144, 160, 176, 192, 208, 224, 240, 255. This results in 13^3 (2197) samples in total. The result of such a sampling is visualized in Figure 2.

These samples are then fed into our Matlab implementations. These implementations use linear interpolation on the subsampled data when performing the color matching. The FGCM algorithm produces a color map M for each projector, and the ICB algorithm generates three independent LUTs for each projector.

After the color maps have been calculated, we apply them and re-measure the color gamuts. Our color map M is applied as detailed in Section 3.4 and the ICB output is applied by loading it into the per-channel Look-Up Tables of the graphics cards. We measured the color matched projectors again using the Sequel Chroma IV colorimeter, subsampling with a 32 increment for each of the input RGB values resulting in 9^3 (729) samples in total.

We use the sampled data from the color matched projectors to generate some error metrics. Our primary metric is the average deviation of the XYZ values of a test color from its average. This can be described as follows. Consider a color matching done over n projectors, and m test colors $c_i = (r_i, g_i, b_i)$, $i = 1, \ldots, m$. Let S_j be the set of measured XYZ values for all test colors on projector j, where $j = 1, \ldots, n$.

$$S_j = \{s_{ij} = (X_{ij}, Y_{ij}, Z_{ij}) = (F_j \circ M_j)(r_i, g_i, b_i) | i = 1, \ldots, m\}.$$

First we define the average response of a test color as

$$\bar{s_i} = \frac{1}{n} \sum_{j=1}^{n} s_{ij}.$$

Then the deviation from the average is

$$E_i = \frac{1}{n} \sum_{j=1}^{n} |s_{ij} - \bar{s_i}|.$$

We can then normalize this set to obtain the percentage deviation at each sample point as

$$e_i = E_i / |\bar{s_i}|$$

And finally we can derive the average of the deviations as a unified metric.

$$\bar{E} = \frac{1}{m} \sum_{i=1}^{m} E_i$$

$$\bar{e} = \frac{1}{m} \sum_{i=1}^{m} e_i$$

In Section 4.2 when refer to E_i and e_i as the absolute error and percentage error respectively for a test color c_i. When we give an overall error metric we are referring to either $\bar{E}$ or $\bar{e}$ as the average deviation from the average expressed as an absolute value or percentage.

4.2. Results

We compare our FGCM algorithm against the ICB algorithm for two test cases: a uniform array of DLP projectors, and a mixed array of DLP and LCD projectors. These cases are presented in Sections 4.2.1 and 4.2.2.

In these cases, the overall color matching error will be a summation of three types of error: measurement consistency error, rounding error, and algorithmic error. Measurement

consistency error is error introduced by the accuracy limitations of the colorimeter and by temporal and spatial variations in measuring the system. Rounding error is the error introduced when rounding a floating point mapping function into an integer range of $[0,255]$. Algorithmic error is the accuracy limitations of a particular color matching algorithm. In our FGCM system, this error is mostly due to interpolation error. We estimate the measurement consistency error to be on the order of 0.4%, based on measuring one projector multiple times. The rounding error is estimated at 0.3%. Since we must measure the system twice, once to find the response of the system, and once to measure the result of color matching, we expect the non-algorithmic error to be on the order of $(0.4^2+0.3^2+0.4^2)^{\frac{1}{2}} = 0.64\%$. So the numbers reported in Sections 4.2.1 and 4.2.2 will contain an inherent system error of about 0.6%.

4.2.1. Case 1: Uniform DLP Projector Array

For our first test case we compare the results of color balancing on four Compaq MP1800 DLP projectors using both our FGCM algorithm and the ICB algorithm. These DLP projectors exhibit the characteristic white enhancement non-additive gamuts as shown in Figure 2. As described in Section 2, non-additive gamuts make it difficult to match the projector colors with an ICB approach. We present the results of our tests in both figure and table form.

Figure 4 visualizes the results of these transformations. Figure 4a–4c show an outline of the color gamut of the four projectors. The six faces of the gamut are created from the 8 sample points (R,G,B,C,Y,M,K,W). The four projectors are represented by the four different line style plots. Figure 4a shows the gamuts after the FGCM algorithm has been applied. Notice that the RGB channels match closely and the white points are well aligned. Figure 4b shows the projector gamuts after ICB is performed. Notice that the red, green and blue channels match well, but there is still a large discrepancy in the white points due to the non-additive gamut. Figure 4c shows the uncalibrated color gamut of the 4 projectors. Notice how the white points are stretched due to white enhancement and that none of the RGB channels or the white points match. Figure 4d–4f show the respective CIE *x*-*y* plot for the (R,G,B,C,Y,M,W) colors. One thing to notice is that even in the uncalibrated Figure 4f the red, green and blue chromaticity values match well among the four projectors. This is because they are of the same model and built within certain tolerances. But the CYMW points are not well aligned due to the non-additive gamut. As can be seen in Figure 4e, ICB does not bring CYMW into alignment. Only Figure 4d which shows FGCM was able to align the CYMW points.

Tables 1 and 2 show the color matching error among the projectors. Table 1 shows the percentage error e_i for 8 test colors (R,G,B,C,Y,M,K,W) and the overall percentage error $\bar{e}$ in the row labeled "total". Table 2 shows the same results

Table 1: *The percentage error (e_i) for solid colors, and the overall percentage error ($\bar{e}$). (All DLP Projectors)*

Color	FGCM	ICB	None
Red (R)	0.75%	1.78%	10.22%
Green (G)	0.77%	1.71%	7.36%
Blue (B)	1.20%	2.38%	10.75%
Cyan (C)	0.86%	2.41%	9.54%
Magenta (M)	1.61%	5.95%	11.02%
Yellow (Y)	1.14%	3.26%	8.64%
Black (K)	3.26%	19.48%	15.59%
White (W)	1.11%	8.95%	10.40%
Total	1.47%	3.40%	11.12%

Table 2: *The absolute error (E_i) for solid colors, and the overall absolute error ($\bar{E}$). (All DLP projectors)*

Color	FGCM	ICB	None
Red (R)	0.229	0.509	3.621
Green (G)	0.770	1.713	7.880
Blue (B)	1.313	2.599	13.256
Cyan (C)	1.435	4.103	18.073
Magenta (M)	1.977	7.230	15.875
Yellow (Y)	1.453	4.003	12.556
Black (K)	0.118	0.207	0.123
White (W)	2.869	21.272	33.237
Total	0.984	2.418	9.710

but gives the absolute error value E_i. Each table shows three columns. The first column labeled "FGCM" is the results from our full-gamut matching algorithm. The second column "ICB" is for the independent channel balancing algorithm and "None" is when no color matching is done. As can be seen, our algorithm has a 1.47% error overall compared to 3.40% for channel balance and 11.12% for no correction. But the effect of gamut matching really becomes obvious near the white point where FGCM has a 1.11% error compared to 8.95% for ICB.

4.2.2. Case 2: Mixed DLP and LCD Projector Array

Our second test case consists of a mixed array of one DLP projector and one LCD projector. We used a Compaq MP1800 DLP projector and a Toshiba TLP511U LCD projector. Mixed arrays of projectors are challenging to color match because the chromaticity of the RGB primaries are likely to be different compared to projectors of a single brand or model. This is true in our test case as can be seen in Figure 5f which shows a CIE *x*-*y* plot of the two projectors. Notice that the chromaticity values of the RGB primaries vary substantially between the two models of projectors.

We again apply our FGCM algorithm and the ICB algorithm to this array of projectors. We present results in the same format as those presented in Section 4.2.1. Figure 5a–5c show the gamut plots for the three cases: FGCM, ICB,

Table 3: *The percentage error (e_i) for solid colors, and the overall percentage error ($\bar{e}$). (Mixed DLP/LCD projectors)*

Color	FGCM	ICB	None
Red (R)	1.33%	5.28%	17.03%
Green (G)	0.64%	3.90%	2.37%
Blue (B)	1.28%	8.25%	15.42%
Cyan (C)	0.82%	5.49%	11.20%
Magenta (M)	0.67%	7.46%	19.38%
Yellow (Y)	1.19%	2.66%	10.09%
Black (K)	11.93%	40.02%	50.22%
White (W)	0.74%	8.28%	32.54%
Total	1.27%	6.21%	12.95%

Table 4: *The absolute error (E_i) for solid colors, and the overall absolute error ($\bar{E}$). (Mixed DLP/LCD projectors)*

Color	FGCM	ICB	None
Red (R)	0.444	1.683	6.269
Green (G)	0.739	4.485	2.821
Blue (B)	1.465	8.703	19.722
Cyan (C)	1.458	9.524	22.419
Magenta (M)	0.827	8.587	28.030
Yellow (Y)	1.625	3.671	15.609
Black (K)	0.836	1.490	0.795
White (W)	1.409	16.821	93.586
Total	1.007	4.435	11.914

and uncalibrated. Figure 5d–5f show the three corresponding CIE *x-y* plots. Notice from Figure 5a that FGCM is able to align the two color gamuts quite well. However ICB, Figure 5b, has significant error even for the RGB colors and the error becomes worse at the white point. The CIE *x-y* Figure 5d–5f show that the uncalibrated projectors have significantly different RGB chromaticity which the ICB algorithm is unable to accommodate. The FGCM algorithm is able to align these points, as seen in Figure 5d.

Tables 3 and 4 show the color matching error values reported as e_i and E_i just as in Section 4.2.1. In this case FGCM is able to achieve a 1.27% overall error compared to 6.21% for ICB and 12.95% for uncalibrated.

4.2.3. Performance Results

We applied our gamut matching algorithm on two of the latest commodity graphics cards: the ATI Radeon 9700 Pro and the Leadtek GeForce4 Ti4600. We tested the performance on an image viewing application we use for our tiled display. Two test PC's are used. The first PC has a 550 MHz Pentium III processor with the GeForce4 card. The second PC has a 3.06 GHz Pentium 4 processor with the ATI Radeon card. Four different shaders are used:

A applies only the geometric alignment
B applies the geometric alignment with an alpha mask

Table 5: *Performance of Image Viewer with Different Pixel Shaders (in Frames Per Second)*

Platform	A	B	C	D
550 MHz P3/GeForce4	22.9	22.6	22.3	22.1
3.06 GHz P4/Radeon	86.4	86.5	86.4	86.4

C applies the geometric alignment with the color map
D applies the geometric alignment, alpha mask and the color map, as described in Section 3.4

The frame rate of the image viewer application is shown in Table 5. It is clear that there is no significant performance hit on either card from applying FGCM.

5. Conclusion and Future Work

Color balancing is critical for the seamless appearance of tiled display walls. Tiled displays composed of projectors from different vendors (mixed arrays) or DLP projectors can present particular challenges when color matching because of chromaticity variation in the red, green and blue channels or due to white enhancement. We have proposed and implemented a non-parametric gamut matching algorithm that can be applied in real time on the latest commodity graphics cards. Our algorithm is able to achieve a measured color uniformity of 1.47% overall compared to 3.40% for an ICB algorithm on a DLP projector array. For mixed DLP/LCD projector arrays, our algorithm is able to outperform the ICB algorithm by a factor of 5, reducing the overall average error from 6.21% to 1.27%.

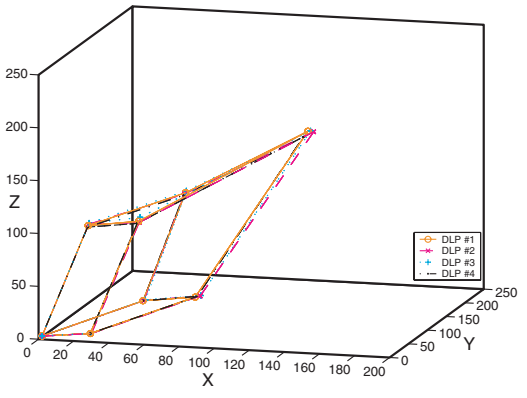
Our future work is to investigate the proper integration of luminance balancing and color matching. Using an alpha mask for luminance balancing implicitly requires a linear color transfer function and a 1.0 gamma value. How to apply the alpha mask while preserving the projector gamma and a non-parametric color mapping still represents a challenge.

6. Acknowledgements

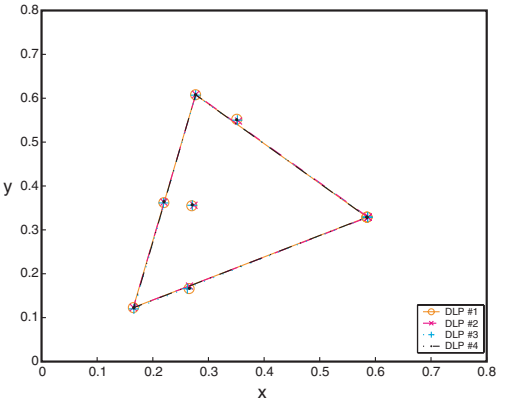
The Princeton Scalable Display Wall project is supported in part by Department of Energy grant DE-FC02-99ER25387, by NSF Infastructure Grant EIA-0101247, by NSF Next Generation Software Grant ANI-9906704, by NCSA Grant ACI-9619019 (through NSF), by Intel Research Council, and by Intel Technology 2000 equipment grant. Han Chen is supported in part by a Gordon Wu Fellowship.

References

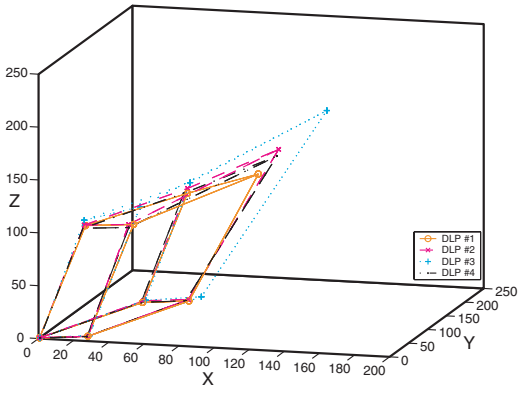
1. H. Chen, R. Sukthankar, G. Wallace, and K. Li. Scalable alignment of large-format multi-projector displays using camera homography trees. In *Proceedings of IEEE Visualization*, 2002.



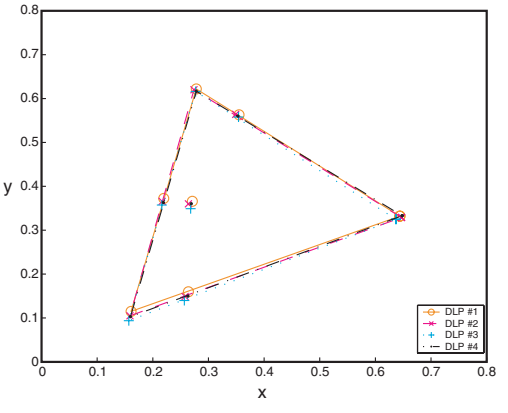
a) FGCM gamut plot for DLP array



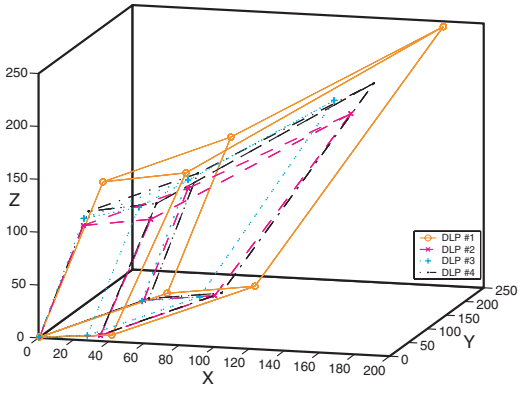
d) FGCM CIE *x-y* plot for DLP array



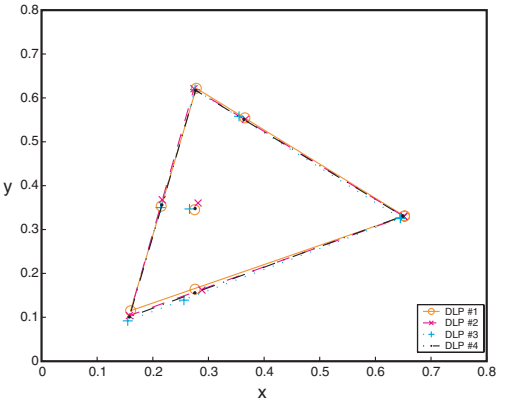
b) ICB gamut plot for DLP array



e) ICB CIE *x-y* plot for DLP array

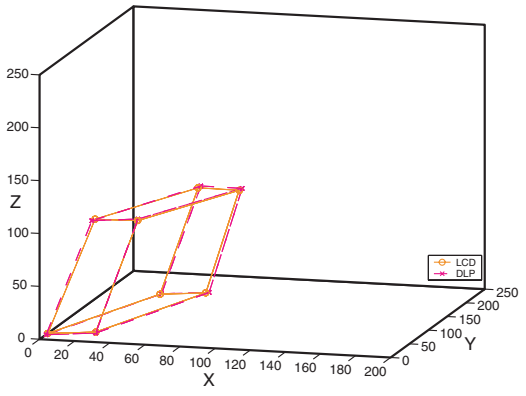


c) Uncalibrated gamut plot for DLP array

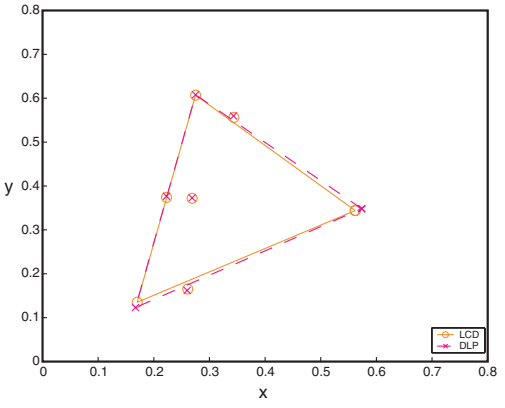


f) Uncalibrated CIE *x-y* plot for DLP array

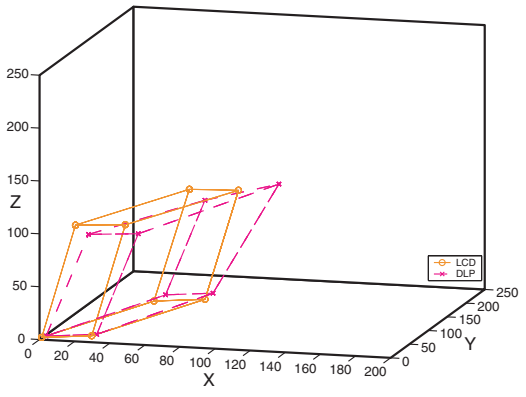
Figure 4: *The measured XYZ gamut plots and CIE x-y plots from FGCM, ICB, and the uncalibrated system for test case 1, where four DLP projectors are used.*



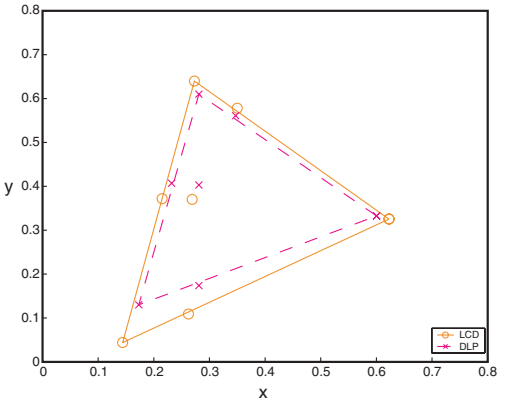
a) FGCM gamut plot for mixed DLP/LCD array



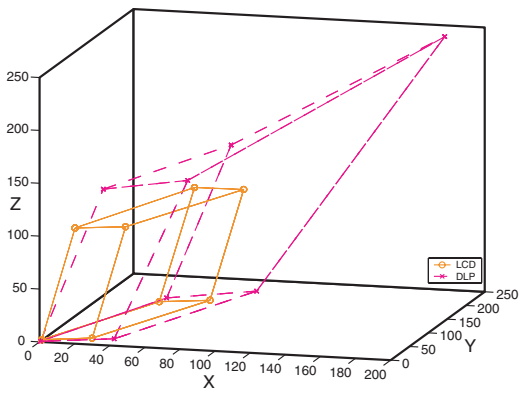
d) FGCM CIE *x-y* plot for mixed DLP/LCD array



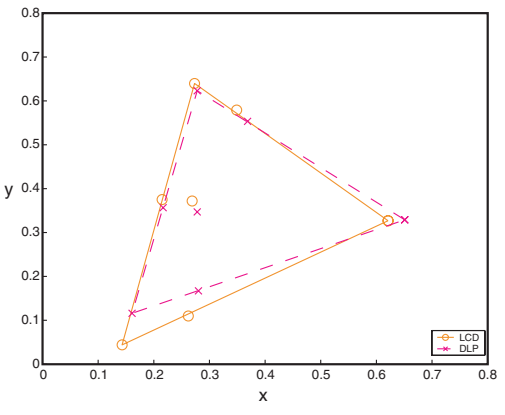
b) ICB gamut plot for mixed DLP/LCD array



e) ICB CIE *x-y* plot for mixed DLP/LCD array



c) Uncalibrated gamut plot for mixed DLP/LCD array



f) Uncalibrated CIE *x-y* plot for mixed DLP/LCD array

Figure 5: *The measured XYZ gamut plots and CIE x-y plots from FGCM, ICB, and the uncalibrated system for test case 2, where mixed DLP and LCD projectors are used.*

2. Y. Chen, D. Clark, A. Finkelstein, T. Housel, and K. Li. Automatic alignment of high-resolution multi-projector display using an uncalibrated camera. In *Proceedings of IEEE Visualization*, 2000.

3. T. Funkhouser and K. Li. Large format displays. *IEEE Computer Graphics and Applications*, 20(4), 2000. Guest editor introduction to special issue.

4. W. Kunzman and G. Pettitt. White enhancement for color sequential dlp. In *SID Conference Proceedings*, 1998.

5. A. Majumder, Z. He, H. Towles, and G. Welch. Achieving color uniformity across multiprojector displays. In *Proceedings of IEEE Visualization*, 2000.

6. A. Majumder and R. Stevens. Lam: Luminance attenuation map for photometric uniformity across a projection based display. In *ACM Virtual Reality and Software Technology*, 2002.

7. R. Raskar, M. Brown, R. Yang, W. Chen, G. Welch, H. Towles, B. Seales, and H. Fuchs. Multi-projector displays using camera-based registration. In *Proceedings of IEEE Visualization*, 1999.

8. M. C. Stone. Color and brightness appearance issues for tiled displays. *IEEE Computer Graphics and Applications*, September 2001.

9. M. C. Stone. Color balancing experimental projection displays. In *9th IS&t/SID Color Imaging Conference*, April 2001.

10. R. Yang, D. Gotz, J. Hensley, H. Towles, and M. Brown. Pixelflex: A reconfigurable multi-projector display system. In *Proceedings of IEEE Visualization*, 2001.