A Paradigm for color gamut mapping of pictorial images

Gustav J. Braun

Follow this and additional works at: https://scholarworks.rit.edu/theses

Recommended Citation
A PARADIGM FOR COLOR GAMUT MAPPING OF PICTORIAL IMAGES

by

Gustav J. Braun

M.S. Rochester Institute of Technology

(1992)

A thesis submitted in partial fulfillment of the requirements for the degree of Ph.D. in the Chester F. Carlson Center for Imaging Science in the College of Science of the Rochester Institute of Technology

October 1999

Signature of the Author

Accepted by Henry E. Rhody
Coordinator, Ph.D. Degree Program
Ph.D. DEGREE THESIS

The Ph.D. Degree Thesis of Gustav J. Braun has been examined and approved by the thesis committee as satisfactory for the thesis requirement for the Ph.D. degree in Imaging Science.

Dr. Mark D. Fairchild

Dr. Roy S. Berns

Dr. Raja Balasubramanian

Dr. Eriko Miyahara

10/28/99

Date
I, Gustav J. Braun, hereby grant permission to Wallace Memorial Library of R.I.T to reproduce my thesis in whole or in part. Any reproduction will not be for commercial use or profit.

Gustav J. Braun

10/28/99

Date
A PARADIGM FOR COLOR GAMUT MAPPING OF PICTORIAL IMAGES

by

Gustav J. Braun

Submitted to the Center for Imaging Science in partial fulfillment of the requirements for the Ph.D. Degree at the Rochester Institute of Technology

October 1999

ABSTRACT

In this thesis, a paradigm was generated for color gamut mapping of pictorial images. This involved the development and testing of: 1.) a hue-corrected version of the CIELAB color space, 2.) an image-dependent sigmoidal-lightness-rescaling process, 3.) an image-gamut-based chromatic-compression process, and 4.) a gamut-expansion process. This gamut-mapping paradigm was tested against some gamut-mapping strategies published in the literature.

Reproductions generated by gamut mapping in a hue-corrected CIELAB color space more accurately preserved the perceived hue of the original scenes compared to reproductions generated using the CIELAB color space.

The results of three gamut-mapping experiments showed that the contrast-preserving nature of the sigmoidal-lightness-remapping strategy generated gamut-mapped reproductions that were better matches to the originals than reproductions generated using linear-lightness-compression functions.
In addition, chromatic-scaling functions that compressed colors at a higher rate near the gamut surface and less near the achromatic axis produced better matches to the originals than algorithms that performed linear chroma compression throughout color space.

A constrained gamut-expansion process, similar to the inverse of the best gamut-compression process found in this experiment, produced reproductions preferred over an expansion process utilizing unconstrained linear expansion.
DEDICATION

This thesis is dedicated to my loving wife Karen Marie Braun and daughter Anna Katherine Braun.

ACKNOWLEDGEMENTS

I would like to thank Dr. Mark Fairchild for his guidance, support, and friendship. I have learned a lot and grown as a scientist and as a person from his example.

I would like to thank Dr. Karen Braun and Dr. Raja Balasubramanian from Xerox for their guidance and support. They were an integral part of much of this work and made it much stronger.

I thank the faculty, staff, and students of the Center for Imaging Science and the Munsell Color Science Laboratory for their many hours of tedious observations. Special thanks go to David Wyble, Colleen Desimone, and Lisa Reniff for their support over the years I was a part of the Munsell Color Science Laboratory.

Finally, I would like to give thanks to my committee (Dr. Roy Berns, Dr. Raja Balasubramanian, and Dr. Eriko Miyahara) for their support and guidance. Special thanks go to Dr. Ethan Montag for serving this thesis and taking part in the defense.
TABLE OF CONTENTS

1 Introduction  
  1.1 Goals  
  1.2 Gamut Mapping for Color Reproduction  
  1.3 What is a gamut?  
  1.4 What is gamut mapping?  
  1.5 Color Spaces for Gamut Mapping  
  1.6 General Categories of Gamut-Mapping Algorithms  
  1.7 Lightness Mapping  
    1.7.1 Device Based Transformations  
    1.7.2 Image Based Transformations  
  1.8 Chromatic Mapping  
    1.8.1 Role of Chromatic Mapping  
    1.8.2 Clipping Functions  
    1.8.3 Scaling Functions  
    1.8.4 Mapping Directions  
    1.8.5 Device- and Image-Dependent Scaling  
    1.8.6 Image-Gamut Scaling  
  1.9 Regional Mappings  
  1.10 Gamut Expansion  
  1.11 Limitations of Current Algorithms  
    1.11.1 Color Space  
    1.11.2 Lightness Rescaling  
    1.11.3 Chroma Mapping  
    1.11.4 Gamut Expansion  
  1.12 Gamut-Mapping Experiments    

2 Techniques for Gamut Surface Definition and Visualization  
  2.1 Summary  
  2.2 Introduction  
  2.3 Gamuts with Known Connectivity
5.4.7 Print-to-Print Results 163
5.4.8 Conclusions 165
5.5 Recommendation of General Gamut-Mapping Algorithm 165

6 Gamut Expansion Experiment 167
6.1 Summary 167
6.2 Gamut-Expansion Algorithm 168
   6.2.1 Lightness Expansion 169
   6.2.2 Chromatic Expansion 171
   6.2.3 Algorithms Tested 174
   6.2.4 Psychophysical Testing 178
6.3 Experimental Results 182
6.4 Conclusions 186

7 Incorporation of Gamut-Mapping Algorithms into Color-Management Systems 188
7.1 "Smart" Color Profiles 189
7.2 "Smart" Color-Management Modules 190
7.3 Hybrid Profiles and Custom CMM 192

8 Conclusions and Recommendations 194

9 References 201

Appendices
LIST OF TABLES

Table 2-1. Results of error analysis. Comparison of mountain-range predicted gamut surface to gamut surface predicted by CRT GOG model. 65

Table 5-1. Description of the gamut-mapping algorithms tested throughout this study. 132

Table 5-2. Description of the gamut-mapping algorithms tested in the print-to-print experiment. 147

Table 5-3. Description of the gamut-mapping algorithms tested in the print-to-print experiment. 159

Table 6-1. Key to the gamut-expansion cases used in this study. 176
LIST OF FIGURES

Figure 1-1. Cross-media color reproduction process 3
Figure 1-2. Illustration of the gamut differences between two imaging devices. 5
Figure 1-3. Hung and Berns lines of constant perceived hue plotted in CIELAB. 7
Figure 1-4. Hung and Berns (1995) lines of constant perceived hue in the CIECAM97s color space. 8
Figure 1-5. An illustration of various lightness rescaling processes. 14
Figure 1-6. Illustration of the homomorphic filtering process ... 16
Figure 1-7. Illustration of a high-pass filter that can be tuned for dynamic range rescaling as well as contrast enhancement. 18
Figure 1-8. Illustration of the chroma-dependent weighting function applied to the lightness scaling function used by Morovic and Luo (1998). 21
Figure 1-9. Illustration of gamut-compression and gamut-expansion regions. 22
Figure 1-10. Illustration of four typical chromatic clipping functions ... 24
Figure 1-11. Illustrations of typical artifacts resulting from using clipping algorithms 26
Figure 1-12. Graphical illustration of clipping (black) and the four main scaling functions discussed in the literature ... 29
Figure 1-13. Illustration of common scaling directions given in the literature. 34
Figure 1-14. Parameters need to set the scaling functions. 36

Figure 2-1. Illustration of a line-gamut boundary plot and the gamut cusp point ... 52
Figure 2-2. Projection and triangulation of uniform RGB cube vertices into L*ab plane. 55
Figure 2-3. Periodic replication of the measured RGB cube data to insure connectivity between h_ab=0° and h_ab=360°. 56
Figure 2-4. Illustration of triangular linear interpolation. 56
Figure 2-5. L*C_ab*h_ab representation of a CRT gamut generated using triangulation and interpolation. 58
Figure 2-6. Mountain-range representation of the L*C_ab*h_ab CRT gamut ... 58
<table>
<thead>
<tr>
<th>Figure 2-7.</th>
<th>Illustration of a slice taken from the mountain-range representation of CRT gamut.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 2-8.</td>
<td>Typical CIELAB &quot;wire-frame&quot; representation of the triangulated and interpolated mountain-range gamut of a monitor.</td>
</tr>
<tr>
<td>Figure 2-9.</td>
<td>Mountain-range gamut of a monitor with simulated measurement and print-to-print variability.</td>
</tr>
<tr>
<td>Figure 2-10.</td>
<td>Noise-filtered mountain-range representation of the gamut shown in Figure 2-9.</td>
</tr>
<tr>
<td>Figure 2-11.</td>
<td>Histogram of ΔE_{ab}^* errors between GOG modeled points and points estimated from the mountain-range gamut.</td>
</tr>
<tr>
<td>Figure 2-12.</td>
<td>CIELAB error vector plots for sample points whose ΔE_{ab}^* was greater than 5.0.</td>
</tr>
<tr>
<td>Figure 2-13.</td>
<td>Comparison of two gamuts using a line-gamut boundary representation.</td>
</tr>
<tr>
<td>Figure 2-14.</td>
<td>Differences between a monitor gamut and a thermal printer gamut shown as a surface plot.</td>
</tr>
<tr>
<td>Figure 2-15</td>
<td>Differences between a monitor gamut and a thermal printer gamut shown as a contour plot.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Figure 3-1.</th>
<th>Extrapolation of Hung and Berns data out to C_{ab}^* = 150.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 3-2.</td>
<td>Twelve gridlines generated from the Hung and Berns data set. These gridlines are used to populate the 2D LUT.</td>
</tr>
<tr>
<td>Figure 3-3.</td>
<td>Samplings from the forward hue-correction transformation every 8 degrees in hue angle. Defines the transformation from CIELAB to hue-corrected CIELAB.</td>
</tr>
<tr>
<td>Figure 3-4.</td>
<td>Samplings from the Inverse hue-correction transformation every 8 degrees. Defines the transformation from hue-corrected CIELAB to CIELAB.</td>
</tr>
<tr>
<td>Figure 3-5.</td>
<td>Interval scales for the 15 hue leaves.</td>
</tr>
<tr>
<td>Figure 3-6.</td>
<td>Interval scale pooled over all images and gamut mapping routines tested.</td>
</tr>
<tr>
<td>Figure 3-7.</td>
<td>Interval scale for images that had predominantly red features that were gamut mapped.</td>
</tr>
</tbody>
</table>
Figure 3-8. Interval scale results for the images that contained predominantly blue features.

Figure 3-9. Interval scale results for the image that contained mixed colored features that were gamut mapped.

Figure 3-10. Forward hue-correction LUT with only correction in the "blue" region generated from the Hung and Berns (1995) data set.

Figure 3-11. Inverse hue-correction LUT with only correction in the "blue" region generated from the Hung and Berns (1995) data set.

Figure 4-1a,b. Family of sigmoidal contrast enhancement functions that have equal $\Sigma$ parameters and varying $x_0$ parameters …

Figure 4-2a-f. Lightness images for the six test images used in the user adjustment experiments …

Figure 4-3a-f. Lightness histograms (bars) and cumulative-lightness histograms (solid gray patches) for the six reference images …

Figure 4-4. User interface for adjustment experiment.

Figure 4-5a-f. Sigmoidal parameter curves from the adjustment experiment.

Figure 4-6. The solid lines represent the $x_0$ and $\Sigma$ parameters resulting from the Phase 3 visual experiment …

Figure 4-7a-c. Optimal $x_0$ and $\Sigma$ parameters as a function of $L^*_{\text{minOut}}$ for the test images sorted into the three image lightness-classes.

Figure 4-8a,b. Final $x_0$ and $\Sigma$ parameter curves for the high, normal, and low-lightness classes.

Figure 4-9a-c. Sigmoidal contrast enhancement remapping functions for high (a), normal (b), and low (c) lightness-classes.

Figure 4-10. Example interpolation of a $x_0$ parameter for an input image with its 75-percent point cumulative-lightness histogram point at $L^*=55$ and an $L^*_{\text{outMin}} = 18$.

Figure 5-1 Illustration of the chromatic-compression functions listed in Table 5-1.

Figure 5-2. Mountain-range representation of the monitor gamut used in the monitor-to-print experiment.
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-3</td>
<td>Mountain-range representation of the HP870Cxi inkjet printer gamut used in the monitor-to-print experiment.</td>
<td>133</td>
</tr>
<tr>
<td>5-4</td>
<td>Contour plot of the difference between the monitor and the HP870Cxi inkjet printer gamuts used in the monitor-to-print experiment.</td>
<td>134</td>
</tr>
<tr>
<td>5-5</td>
<td>Illustration of the interface used to present the images in the monitor-to-print experiment.</td>
<td>136</td>
</tr>
<tr>
<td>5-6</td>
<td>Interval scale results for the monitor-to-print experiment.</td>
<td>139</td>
</tr>
<tr>
<td>5-7</td>
<td>Mountain-range representation of the Xerox Regal MajestiK gamut.</td>
<td>142</td>
</tr>
<tr>
<td>5-8</td>
<td>Mountain-range representation of the Xerox Xpress gamut.</td>
<td>143</td>
</tr>
<tr>
<td>5-9</td>
<td>Mountain-range representation of the Fuji gamut generated from the scanner model.</td>
<td>144</td>
</tr>
<tr>
<td>5-10</td>
<td>Contour plot of the difference between the Fujix gamut and the MajestiK gamut.</td>
<td>145</td>
</tr>
<tr>
<td>5-11</td>
<td>Contour plot of the differences between the Fujix gamut and the Xpress gamut.</td>
<td>146</td>
</tr>
<tr>
<td>5-12</td>
<td>Illustration of the color management path used to create the hardcopy reproductions for the print-to-print pilot experiment.</td>
<td>148</td>
</tr>
<tr>
<td>5-13</td>
<td>Interval scale results from the print-to-print pilot experiment.</td>
<td>151</td>
</tr>
<tr>
<td>5-14</td>
<td>Mountain-range representation of the Xerox Royal MajestiK printer gamut.</td>
<td>155</td>
</tr>
<tr>
<td>5-15</td>
<td>Mountain-range representation of the Xerox Xpress printer gamut.</td>
<td>155</td>
</tr>
<tr>
<td>5-16</td>
<td>Mountain-range representation of the Fujix printer gamut generated from the printer characterization.</td>
<td>156</td>
</tr>
<tr>
<td>5-17</td>
<td>Contour plot of the difference between the Fujix gamut and the MajestiK gamut.</td>
<td>157</td>
</tr>
<tr>
<td>5-18</td>
<td>Contour plot of the difference between the Fujix gamut and the Xpress gamut.</td>
<td>158</td>
</tr>
<tr>
<td>5-19</td>
<td>Average interval scale across the nine images used in the softproofing experiment.</td>
<td>163</td>
</tr>
<tr>
<td>6-1</td>
<td>Illustration of the compress/expansion (TYPE I/TYP END) regions for a sample source and destination gamut.</td>
<td>169</td>
</tr>
</tbody>
</table>
Figure 6-2. Illustration of the chromatic enhancement function used in gamut-expansion regions. 172

Figure 6-3. Illustration of the gamut expansion constraints placed on the maximum chromatic boost. 174

Figure 6-4. For this hue leaf, the chromatic boost does not exceed 30-percent for any region of the gamut. 175

Figure 6-5. Illustration of the gamut clipping process that was required to insure that no lightness dynamic range expansion was performed for Case 1 and Case 2 ... 178

Figure 6-1. Surface plot of the difference between the gamuts used in the gamut expansion experiments. 179

Figure 6-7. Contour plot of the difference between the monitor gamut and the Fujix gamut used in the gamut-expansion experiment. 180

Figure 6-8. Color management process used in the gamut-expansion experiments. 182

Figure 6-9. Interval scale values for the algorithms tested in the gamut-expansion experiment (averaged across the eight scenes). 183

Figure 8-1. High-level flow chart of the gamut-mapping process. 197

Figure 8-2. Illustration of the lightness-mapping process used in the general-purpose gamut-mapping algorithm. 198

Figure 8-3. Illustration of the chromatic mapping stage of the general-purpose gamut-mapping algorithm. 199

Figure 8-4. Polar gamut-mapping process performed in the "Gamut Map" stage given in Figure 8-3. 200
1 Introduction

1.1 Goals

The goal of this research was to define a general methodology for color gamut mapping of pictorial images. This was accomplished by the following:

- Definition of a reference color space for accurate hue rendition (Section 3). Traditional color spaces used for gamut mapping (namely, CIELAB) have been shown to be non-uniform with respect to perceived hue lines. As such, a hue-linearized modified version of CIELAB was developed and tested for gamut mapping purposes.

- Definition of an image-dependent gamut-mapping strategy that preserves the lightness and chromatic content of the original scenes (Section 4 and 5). This strategy employs image reproduction principles based on the input image data present. The strategy utilizes contrast-boosting lightness rescaling, followed by 2D histogram based chroma-scaling procedures.

- Application of the concepts of the gamut-compression procedures to gamut-expansion. Gamut expansion was approached as the inverse problem of gamut compression. The gamut-expansion functions were generated by inverting the gamut-compression functions. (Section 6)
1.2 Gamut Mapping for Color Reproduction

The process of reproducing an image, rendered on a given source medium, onto a different medium is referred to as cross-media color reproduction. A general processing diagram is presented for cross-media color reproduction in Figure 1-1. In order to maintain the appearance of the original image on the source medium/device on the destination device, at least two factors need to be considered. The first deals with the psychophysical effects of generating visual matches of images that are displayed in different viewing conditions (i.e., differences in white-point, surround, illumination levels, etc.). This topic has been covered by many researchers (see Fairchild (1998) for references) and is not the focus of this thesis.
The second factor that needs to be considered is the difference in color gamut between two devices, which utilize different sets of primaries to generate images. For example, additive systems use red, green, and blue light to generate colors. Subtractive systems use cyan, magenta, yellow, and black inks, dyes, or toners to generate colors. These primaries, in combination with the image processing, define the color gamut of the
device. That is, there will be a different set of colors a monitor is able to generate than a CMYK printer. Overcoming these gamut differences is the focus of this thesis and is what is commonly referred to as color-gamut mapping. The following sections give details of:

1.) Some of the strategies that have been tried to compensate for lack of similarity between device gamuts;
2.) Some color reproduction issues associated with these strategies;
3.) A proposed general-solution for compensating for the differences between gamuts in a color reproduction system.

1.3 What is a gamut?

A gamut represents the set of all colors that a color-reproduction device is physically able to generate. This set has two components that need to be considered. The first is the gamut boundary. The gamut boundary of a device represents the outer-most extent of the device's capabilities in some reference color space. The second is the number of colors that are realizable within the gamut-boundary. Because of quantization in color reproduction systems, such as in digital halftone devices, not all colors that are within a device's boundary are realizable. (Note: In all cases presented in this thesis, the color gamut of a device is assumed to be continuous within the gamut boundary; i.e., no device quantization effects were considered by any of the gamut mapping algorithms.)

The shape and extent of this volume are generally a function of the device primaries and the viewing environment under which the reproductions are observed. Significant
differences can exist between the gamuts produced by color imaging systems that utilize different primaries and viewing environments. The gamuts shown in Figure 1-2 show the differences in the gamut boundary of two imaging devices in the CIELAB L* a* b* color space. The solid gamut is from a thermal printer and the wire-frame gamut is from a monitor display.

Figure 1-2. Illustration of the gamut differences between two imaging devices. The wire-frame gamut is from a monitor display and the solid gamut is from a thermal printer.

In order to perform any gamut mapping operations it is necessary to have an accurate description of the gamuts for all display devices used. A gamut-specification process has been proposed Braun and Fairchild (1997) and is given in Section 2. This process offers a robust procedure for estimating a device’s gamut from measured or modeled data.
1.4 What is gamut mapping?

Gamut mapping involves a series of transformations designed to compensate for the lack of universal intersection between the set of colors that are realizable on an input device and an output device. The gamut-mapping process transforms a point in the source gamut to a realizable color inside the gamut of the output device. The form of this transformation can dramatically impact the quality of the reproduced image. As such, care needs to be used in the design and implementation of gamut-mapping transformations.

1.5 Color Spaces for Gamut Mapping

The hue linearity of a reference color space is critical in color gamut mapping. When a color is reduced in chroma following a line of constant metric hue angle (e.g., CIELAB $h_{ab}$) to fit within a destination gamut, the perceived hue of that color will change if the reference color space is non-linear with respect to hue. This has been shown to be the case for the CIELAB color space (Hung and Berns (1995), Ebner and Fairchild (1998a,b)). Thus, lines/planes of constant metric hue angle do not correspond to lines/planes of constant perceived hue in the CIELAB color space, as shown in Figure 1-3. Some of the strongest non-linearities occur in the red and blue regions of the color space.
Recently, the *Commission Internationale de l' Eclairage* (CIE) has recommended the CIECAM97s color space (CIE (1998)) for describing color appearance and for generating corresponding color matches across changes in viewing conditions. Unfortunately, the CIECAM97s color space is also non-linear with respect to hue. This can be illustrated by plotting the Hung and Berns visual data set in the CIECAM97s color space, Figure 1-4.

*Figure 1-3. Hung and Berns lines of constant perceived hue plotted in CIELAB.*
Recently, McCann (1999) and Marcu (1998) have suggested that gamut mapping be performed in a color space generated from transformations based on the Munsell renotation data. In their processes they use the uniformly spaced Munsell renotation data to generate 3-dimensional look-up tables (LUT) to transform in and out of their respective hue-linearized spaces. The main drawback of their processes is that the Munsell data used to describe the color space was developed from samples that had very
limited chroma ranges. Unfortunately, most of the gamut mapping occurs in high-chroma regions of color space. The data generated by Hung and Berns (1995) and Ebner and Fairchild (1998a) came from a much larger gamut and are thus more appropriate than the data used by Marcu or McCann.

Ebner and Fairchild (1998b) generated a promising new color space called IPT that has improved hue linearity. This space is based on the research performed by Ebner for his Ph.D. thesis (1998). This space was not used in this thesis since it was being generated concurrently with this thesis. In addition, the IPT color space has not yet been thoroughly tested for gamut-mapping experiments.

Based on the widespread use and acceptance of the CIELAB color space in color imaging, two attempts were made to generate hue-linearized versions of CIELAB for color gamut mapping. The first was based on the constant hue-line data from Hung and Berns (1995) and the second was based on a set of constant hue planes from Ebner and Fairchild (1998a). Section 3 details how these visual data were used to create multi-dimensional look-up-tables (LUTs) that linearized the hue of CIELAB color space and provided a sound basis for color gamut mapping. A visual experiment that compared these LUTs for a color gamut mapping application is presented in Section 3.
1.6 General Categories of Gamut-Mapping Algorithms

Gamut mapping algorithms come in a variety of levels of complexity and utility. The least complex algorithms simply clip digital counts in device-dependent space (RGB, CMY, or CMYK) when a requested digital count is beyond the range of a device. Other methods apply global mapping functions in the form of one-dimensional LUT's to map the perceptual attributes of one gamut into the volume of another (Montag and Fairchild (1997), Gentile, et al. (1990), Hoshino and Berns (1993), Pariser (1991), and MacDonald (1993)). Still others divide the gamut and/or the reference color space into regions and apply regional transformations to image data (Wolski, et al. (1994), Spaulding, et al. (1995), Luo and Morovic (1998)). In general, gamut-mapping algorithms can be broken down into four distinct classes:

1.) Clipping of out-of-gamut points to the destination-gamut boundary (Section 1.8.2.)
2.) Global lightness and chromatic-mapping functions (Sections 1.7 and 1.8)
3.) Regional-dependent mapping schemes (Section 1.9)
4.) Gamut-expansion techniques (Section 1.10).

1.7 Lightness Mapping

One of the most important factors in a color-gamut-mapping process for pictorial images is that the final image maintains the lightness integrity of the original scene. Often times
an output device, such as an inkjet or laser printer, does not possess the lightness dynamic range that is present in the original scene (e.g., that of a CRT or a glossy photographic original). When the large input dynamic range is rescaled to fit into the smaller output dynamic range, significant image contrast may be lost. For the purposes of this discussion, *lightness dynamic range* refers to the difference between the lightness of the device white point and black point. *Contrast* is an image-specific quantity that relates to the form of the lightness histogram. An image with a narrow lightness histogram will appear to have low visual contrast. As the lightness histogram is broadened, the image will appear to have higher visual contrast (Gonzalez and Woods (1992), pp. 174). Thus, within a given lightness dynamic range, the perceived contrast of an image can be manipulated by narrowing or broadening its lightness histogram.

1.7.1 Device Based Transformations

Various lightness-remapping schemes have been formulated to account for these dynamic range differences. Linear lightness rescaling is the most common lightness rescaling process cited in the gamut mapping literature (Montag and Fairchild (1997), Morovic and Luo (1997a,b, 1998), Stone and Wallace (1991), Viggiano and Wang (1992), MacDonald and Morovic (1995), Morovic (1998)). In this case, source pixel lightness values are processed through the following linear scaling equation:

\[
L_{\text{out}}^* = \frac{L_{\text{in}}^*}{100} (100 - L_{\text{minOut}}^*) + L_{\text{minOut}}^*
\]  

(1-1)
where \( L^*_{\text{out}}, L^*_{\text{in}}, \) and \( L^*_{\text{minOut}} \) are the mapped lightness, the lightnesses of the source pixels, and the minimum lightness of the destination device, respectively. In general, the source pixels will not have lightnesses that span the entire CIELAB \( L^* \) [0 100] range. As such, a slight modification can be made to the remapping function given in Equation 1-1 where the range of input image data is scaled into the range of the destination device (Montag and Fairchild (1997)). Thus, the source pixel data are first normalized to a full lightness range (i.e., \([L^*_{\text{minIn}} 100] \rightarrow [0 1]) then rescaled into the full range of the destination device (i.e., \([0 1] \rightarrow [L^*_{\text{minOut}} 100]) as shown in Equation 1-2.

\[
L^*_{\text{out}} = \frac{L^*_{\text{in}} - L^*_{\text{minIn}}}{100 - L^*_{\text{minIn}}} (100 - L^*_{\text{minOut}}) + L^*_{\text{minOut}}
\]  

(1-2)

When the \( L^*_{\text{minIn}} \) value is significantly greater than zero, this process more efficiently utilizes the limited dynamic range of the output device. In spite of this modification, the linear lightness remapping process suffers from a global reduction in the perceived lightness contrast and an increase in the mean lightness of the remapped image. When the dynamic range difference between the source and destination devices is significant, output images tend to appear light and often times contain a “milky” or “hazy” appearance in the shadow detail.

Lightness clipping algorithms (hard clipping) can be applied to reduce the loss in perceived lightness contrast obtained when the destination dynamic range is less than the source dynamic range. In this case, all of the source pixel lightnesses that are less than the
destination device's black-point lightness are clipped to that black-point lightness. In general, this process can approximately maintain the mean lightness of the image. The major shortcoming of this process is the potential for significant loss in image texture resulting from the many-to-one mapping in dark regions. For images that contain significant amounts of shadow detail below the $L_{\text{minOut}}^*$ lightness, clipping results in a "flattening" of the shadowed regions. This phenomenon is accentuated as the $L_{\text{minOut}}^*$ increases.

In an effort to overcome the loss in detail associated with hard clipping, soft-clipping procedures have been applied (Hoshino and Berns (1993), Montag and Fairchild (1997)). Examples of some typical soft clipping functions are given in Figure 1-5. In the dark region, the contrast is compressed in the shadow detail, unlike straight hard clipping. As the lightness approaches its maximum, an identity mapping is utilized, $L_{\text{out}}^* = L_{\text{in}}^*$. As a result, the lightness contrast in the shadow region is compressed but the texture remains visible. The two forms of the soft-clipping functions shown in Figure 1-5 are knee compression and soft compression. The soft-compression function increases the rate of compression as the minimum lightness is approached. The knee function applies the same compression rate throughout the compression region (piece-wise linear).

While, soft- and knee-compression rescalings offer more flexibility than hard clipping or linear lightness rescaling, it is difficult to generalize these processes for all image types and dynamic ranges. As the black point of the destination device increases progressively
to higher lightnesses, progressively more low-end compression is needed. The ways to achieve this are by: 1.) performing more compression in the knee-segment or the soft-compression region, or 2.) utilizing a smaller linear remapping region. Each of these options has the drawback of reducing the perceived lightness contrast of the remapped images.

![Graph showing various lightness rescaling processes](image)

Figure 1-5. An illustration of various lightness rescaling processes. Linear compression is given by Equation 1-1. Hard clipping sets all input $L^*$ values less than 30 to an output of $L^*=30$. The knee compression function in this example performs linear compression of input $L^*$ values from [0 60] into the output range of [30 60]. The soft compression function gradually compresses the input $L^*$ values between [0 60] into the output range of [30 60].
1.7.2 Image Based Transformations

1.7.2.1 Homomorphic Filtering

The major problem associated with the device-centric transformations discussed in Section 1.7.1 is that they do not take into account what happens to the image data during the transformations. A study by Meyer and Barth (1989) addressed the contrast loss that results from lightness compression by applying a homomorphic-filtering operation to achieve dynamic-range compression. The basis for their rescaling process was that the image is composed of a low-frequency intensity component and a high-frequency reflectance component. For dynamic range rescaling, it is desirable to compress only the low-frequency component and leave the high-frequency component unchanged. This helps to maintain the contrast in the remapped image. The logarithmic transformation of the homomorphic-filtering process helps to decouple the low-frequency illumination and the high-frequency reflectance components of the image.

Meyer and Barth (1989) gave few details of the transformation. As such, it was not possible to duplicate their process. The transformations given below were inferred from the concepts that they developed and a detailed description of homomorphic filtering found in Gonzalez and Woods (1992). An illustration of the process is given in Figure 1-6.
Figure 1-6. Illustration of the homomorphic filtering process where the input image \( f(x,y) \) is transformed to the output image \( g(x,y) \) by the transformations given in Equations 1-3-1-7.

The theory behind the homomorphic filtering process is as follows. Consider an input image, \( f(x,y) \), to be the product of a low-frequency illumination component, \( i(x,y) \), and a high-frequency reflectance component, \( r(x,y) \), Equation 1-3.

\[
f(x,y) = i(x,y)r(x,y)
\]  \hspace{1cm} (1-3)

The idea of homomorphic filtering to perform the lightness rescaling is that it can be used to separate \( f(x,y) \) into its low- and high-frequency components. This is accomplished by applying the logarithmic transformation to the input scene \( f(x,y) \), Equation 1-4.

\[
z(x,y) = \ln(f(x,y)) = \ln(i(x,y)) + \ln(r(x,y))
\]  \hspace{1cm} (1-4)

Fourier filtering of \( z(x,y) \) can then be used to reduce the dynamic range of the image as well as boost the contrast of the high frequency data. The general filtering functions are given by:

\[
S(u,v) = Z(u,v)H(u,v) = H(u,v)i(u,v) + H(u,v)r(u,v)
\]  \hspace{1cm} (1-5)
where \( S(u,v) \) is the frequency domain representation of the filtered version of the \( z(x,y) \), \( Z(u,v) \) is the Fourier transform of \( z(x,y) \), \( I(u,v) \) and \( R(u,v) \) are the Fourier transforms of \( \ln(i(x,y)) \) and \( \ln(r(x,y)) \), respectively, and \( H(u,v) \) is a high-pass filter. The form of this filter \( (H(u,v)) \) can be represented by:

\[
H(u,v) = 1 - \text{GAUS}(u,v)
\]  

(1-6)

where \( \text{GAUS}(u,v) \) is a 2-dimensional Gaussian function in the spatial-frequency domain. By adjusting the width and the amplitude of \( \text{GAUS}(u,v) \), the spatial frequency characteristics of \( H(u,v) \) can be tuned. Gonzalez and Woods (1992) defined a general circularly-symmetric filter, \( H(u,v) \), that can be used for controlling the dynamic range and the contrast of the filtered image, Figure 1-7. By setting \( \gamma_l < 1 \) and \( \gamma_H > 1 \) this filter performs simultaneous dynamic-range compression and contrast enhancement. The final filtered image takes on the form of:

\[
g(x,y) = \exp(\text{FFT}^{-1}(S(u,v))) = \exp(\text{FFT}^{-1}(I(u,v)H(u,v)) + \text{FFT}^{-1}(R(u,v)H(u,v)))
\]  

(1-7)
Figure 1-7. Illustration of a high-pass filter that can be tuned for dynamic range rescaling as well as contrast enhancement. (Figure was taken from pp. 217 from Digital Image Processing, Gonzalez and Woods, 1992.) The \( D(u, v) \) axis represents radial spatial frequency coordinates and the vertical axis represents the amplitude of the filter \( H(u,v) \).

In order to test this process, various values for \( \gamma_L < 1 \) and \( \gamma_H > 1 \) were tried for a lightness rescaling case from a full L* range [0 100] to a constrained range of [30 100]. In addition to controlling the scaling parameters, the width of the Gaussian filter used to generate \( H(u,v) \) was adjusted. The results of this process were not good. In order to make decent reproductions, the \( \text{GAUS}(u,v) \) went to a delta function. This resulted in a remapping that was given by the following form:

\[
g(x,y) = \gamma'_L f(x,y)^{Y_H}
\]  

(1-8)

where \( \gamma'_L = \exp(-\gamma_L Z(u=0,v=0)) \). The derivation of the filtering process shown by Equation 1-8 is given in Appendix A. Essentially this process can be replaced by a one-dimensional LUT using a power function and a dynamic range scalar. The images mapped through this process were higher contrast but were not good matches to the originals.
1.7.2.2  Histogram Equalization

The process of histogram equalization can be used to increase the contrast of an image. This process is described by Gonzalez and Woods (1992, p. 173). Essentially, equalizing or flattening the lightness histogram of the image increases the contrast of an image. This is accomplished by using the cumulative-lightness histogram of the image as a remapping function. The resultant image is generally much higher contrast than the original.

The process for using this as a lightness rescaling function for gamut mapping is as follows. The original scene is “equalized” by its cumulative-lightness histogram and then linearly rescaled into the destination dynamic range. The increase in contrast associated with the equalization process can be used to help overcome the loss in contrast associated with the lightness rescaling process. This process was tried on several images and abandoned based on the quality of the reproductions. In all cases, the remapped scenes were much higher contrast than the originals.

1.7.2.3  Modified Histogram Equalization

A modification can be made to the standard histogram-equalization process that tempers the amount of contrast enhancement (Hains (1998)). This process first scales the image histogram by a logarithmic transformation. The image is then remapped using a LUT generated from the cumulative log-histogram. The logarithmic transformation reduces the variation in the image histogram and results in a smoother transformation. This process was performed on several images, but was abandoned because it did not produce images
that were close matches to the originals. In most cases, these images were higher contrast than obtained using straight linear rescaling but lower contrast than the original scenes.

1.7.2.4 Chroma-Weighted Gaussian Linear Scaling (GCUSP)

Recently, Morovic and Luo (1997a) performed a series of gamut-mapping experiments aimed at developing a “universal approach” to color gamut mapping. In their studies they concluded that the lightness-compression rate should be less for high-chroma colors than for low-chroma colors. This process helped to maintain the chroma of higher chroma colors during the gamut-mapping process. In order to do this, they generated a chroma-weighted linear scaling based on a Gaussian-like function. They used a linear-lightness scaling equation that was scaled by a chroma-dependent function given by:

\[ p_c = 1 - \frac{\left( \frac{C^*_{\text{input}}}{5 \times 10^5} \right)^3}{\left( \frac{C^*_{\text{input}}}{5 \times 10^5} \right)^3 + 5 \times 10^5} \]  \hspace{1cm} (1-9)

Therefore, their final lightness scaling is given by:

\[ L^*_{\text{output}} = (1 - p_c) L^*_{\text{input}} + p_c \left( L^*_{r_{\text{max}}} - L^*_{o_{\text{max}}} \right) \left( \frac{L^*_{r_{\text{max}}} - L^*_{r_{\text{min}}}}{L^*_{o_{\text{max}}} - L^*_{o_{\text{min}}}} \right) \]  \hspace{1cm} (1-10)

where \( L^*_{r_{\text{max}}}, L^*_{r_{\text{min}}}, L^*_{o_{\text{min}}}, \) and \( L^*_{o_{\text{max}}} \) are the maximums and minimums of the source, \( r \), and destination, \( o \), gamuts respectively. A plot of the chroma-dependent weighting function is shown in Figure 1-8. Based on findings of Morovic and Luo (1998) extensive testing of the GCUSP algorithm was performed in Section 5 of this thesis.
Figure 1-8. Illustration of the chroma-dependent weighting function applied to the lightness scaling function used by Morovic and Luo (1998). (Note: This figure was taken directly from Morovic (1998) Figure 7.2.1, pp. 163.)

1.8 Chromatic Mapping

1.8.1 Role of Chromatic Mapping

The role of chromatic mapping is to insure that all of the points outside the output device gamut are moved inside that gamut before they are converted to device coordinates for output. The different regions highlighted in Figure 1-9 were referred to as Type I and Type II gamut-mapping regions by Gentile, et al. (1990). The Type I regions are the compression regions and must be compressed in order to reproduce any possible image from within the source gamut. This has been the focus of nearly all of the gamut mapping literature. The Type II regions are areas in color space where the destination gamut is larger than the source gamut. In these regions, the dynamic range of an image could be expanded to produce higher contrast and more chromatic images.
In his thesis, Ján Morovic (1998) gave an impressive chronological accounting of the recent history of gamut-mapping algorithms. Anyone interested in gamut mapping should read this to familiarize themselves with the field. The bulk of that literature review section was submitted by Morovic and Luo (1999) to *Color Research and Application* as a part of the charter of the CIE Technical Committee 8-3. This Technical Committee is currently working to standardize a color-gamut-mapping strategy. The following sections group some of the most important gamut-mapping studies into categories and discuss the advantages and disadvantages of each.
1.8.2 Clipping Functions

Clipping has traditionally been one of the most widely used gamut-mapping algorithms in the literature (Sara (1984), Gentile, et al. (1990), Pariser (1991), Ito and Katoh (1999), Katoh and Ito (1996), Montag and Fairchild (1998)). There are several features that make clipping useful for gamut mapping. The first is that it is generally easy to implement. A given point is clipped by moving that point to the surface of the destination gamut according to some rule. This rule may be to follow the minimum distance path to the gamut surface, Figure 1-10a. Other simple clipping rules are shown in Figure 1-10b-d. Each of the cases shown in Figure 1-10 has its own particular benefits and drawbacks.
Another reason chromatic clipping is very popular is that it is very general. The process acts independently of source and image gamut. Thus, it can be used for any type of input data regardless of origin. Clipping functions are easily coded into multi-dimensional LUTs, making them prime candidates for color-management systems.
Clipping algorithms are also image-dependent in that an image that has very few highly chromatic features is not affected much by clipping. Images with many chromatic features are affected more. By leaving the low-chroma, in-gamut features unchanged, the near neutral colors of the image are affected less than by functions that scale in-gamut as well as out-of-gamut data (Section 1.8.3) This may be important in the reproduction of skin-tones and other lower-chroma features.

The downside of clipping algorithms is that they can create quantization and contouring artifacts in the mapped image. These phenomena are shown in Figure 1-11a,b. Feature (A) shown, in Figure 1-11a, is mapped to a very narrow region of color space on the surface of the gamut. The many-to-one nature of clipping algorithms often results in loss of lightness and chromatic contrast. Feature (B) in Figure 1-11a is partially in-gamut and partially out-of-gamut. The clipping process maps all of the out-of-gamut pixels to the gamut surface, leaving the in-gamut pixels unchanged. If the points in Feature (B) came from a smoothly varying object such as a lightness and chroma gradient, then the clipping process would generate a hard contour line in the smoothly varying input object. In an extreme case, clipping can result in an entire object being mapping to a single point in color space. In the case of minimum-distance clippings an entire feature may be mapped to the tip of a gamut slice, Figure 1-11b.
Figure 1-11. Illustrations of typical artifacts resulting from using clipping algorithms: (a.) The entire input feature (A) is mapped to the gamut surface. Some lightness contrast and nearly all of the chromatic contrast in the feature is lost. Feature (B) is distorted by the clipping process because it crossed the gamut boundary. (b.) In the minimum distance to the gamut surface, all of the points in the feature are mapped a single point in color space.

In some respects, the clipping directions have been formulated based on fairly obvious criteria. For example, the minimum-distance-clipping algorithm changes the colorimetry of the image the least of all the clipping algorithms. The constant-lightness chroma-clipping process preserves the lightness tone reproduction of the scene. The centroid-clipping algorithm gives a common point to map toward for all hue angles and cusp-point clipping maps toward the “fattest” point of the gamut for each hue angle.

Recently, several researchers have tried to determine optimal clipping directions from psychophysical experimentation (Katoh and Ito (1996), Ito and Katoh (1999), Ebner and
Fairchild (1997)). These studies determined clipping directions by having observers adjust colors or optimize direction functions that produce minimum visual differences.

For simple patches and clip-art images, Ebner and Fairchild (1997) tried to quantify the direction observers adjusted out-of-gamut colors onto the surface of a reference gamut. Their analysis consisted of fitting a vector component model to the user-adjusted data. The vector analysis was optimized such that the user-adjusted vectors were replicated as weighted sums of component vectors. The three-component vectors used were a minimum $\Delta E_{ab}^*$ vector, a centroid vector, and a chroma-clipping vector that preserved lightness and hue. This analysis resulted in a parametric vector calculation that could be used to define the optimal direction of compression for simple-field object colors. Recent work by Katoh and Ito (1999) has shown that ratios of 1:2:1 and 1:2:2 for compression of lightness, chroma, and hue has produced encouraging results for mapping out-of-gamut colors onto the surface of a reference gamut. It is hoped that this type of research leads to more accurate clipping directions for applications that support these types of transformations (e.g., computer graphics, business graphics, etc.).
1.8.3 Scaling Functions

In order to counteract the disadvantages of clipping algorithms, many different chromatic-scaling functions have been utilized. For discussion purposes, these functions have been summarized by the following four functions, Figure 1-12:

1. Soft-clipping compression
2. 3-piece linear compression
3. Knee function compression
4. Linear compression
These four compression functions cover most of the cases that have been presented in literature. The most straightforward form of chromatic compression is linear scaling. Linear chromatic scaling can be generalized by the following form:

\[
output = \text{input} \left( \frac{\text{Max}_{\text{output}}}{\text{Max}_{\text{input}}} \right),
\]

(1-11)
where $\text{Max}_{\text{input}}$ and $\text{Max}_{\text{output}}$ are the maximum respective input and output ranges along
the given scaling line. Linear scaling preserves the relative relationship of the colors
along the line. It does not produce contours or result in the quantization of points. It does,
however, unnecessarily reduce the chromatic contrast of the low-chroma features as well
as high-chroma features. This has a more significant effect on the scene than just
reducing the chromatic nature of the highly chromatic features (Montag and Fairchild
1997, Gentile, et al. (1990), Braun and Fairchild (1999b,d)).

In order to correct the global loss in chromatic contrast associated with linear-chromatic
compression, researchers have utilized non-uniform chromatic compression functions.
All of these functions have a common form in that they compress more in the highly
chromatic regions and less (or not at all) in the low-chroma regions. The most
straightforward form of non-uniform chromatic compression is given by the knee-
function scaling (Gentile, et al. (1990), Montag and Fairchild (1997), Hoshino and Berns
(1993), Braun and Fairchild (1999b,d)). This function maintains the chromatic content of
the colors up to the knee point, and chromatic compression occurs from the knee-point to
the maximum input point (most often the source gamut boundary). This function can be
generalized by the following two-piece linear equation:

$$
\text{output} = \begin{cases} 
\text{input}, & 0 \leq \text{input} \leq K \\
\left( \frac{\text{Max}_{\text{output}} - K}{\text{Max}_{\text{input}} - K} \right) \text{input} + K \left( 1 - \frac{\text{Max}_{\text{output}} - K}{\text{Max}_{\text{input}} - K} \right), & K < \text{input} \leq \text{Max}_{\text{input}} 
\end{cases} 
$$

(1-12)
where $K$ is the value of the knee point and $\text{Max}_{\text{input}}$ and $\text{Max}_{\text{output}}$ are the maximum input and output ranges, respectively, along the given scaling line. By adjusting the knee point, $K$, to different values in the range of $[0 \text{ Max}_{\text{output}}]$, the knee-function relationship can produce scaling functions from linear-chromatic compression through chromatic clipping. The knee-function rescaling process offers more flexibility than linear-chromatic compression to maintain the chromatic content of the low chromatic features.

The soft-clipping compression function is a smooth alternative to the knee function. The exact forms of the soft-clipping compression vary from study to study, but they all have the same general form (Herzog and Muller (1997), Hoshino and Berns (1993), Stone and Wallace (1991)). The idea with soft clipping is that more compression is performed near the edges of the gamut rather than on points that are within the destination gamut. Unlike, the piece-wise linear knee function, the soft-clipping functions smoothly increase the compression rate from some point out to the gamut surface. These functions perform very similar to knee functions.

A hybrid between straight clipping and a knee function is given by the three-piece linear function given by Gentile, *et al.* (1990), shown in Equation 1-13. In this relationship $K$ and $K'$ are the reference points shown in Figure 1-12 and $\text{Max}_{\text{input}}$ and $\text{Max}_{\text{output}}$ are the maximum input and output ranges, respectively, along the given scaling line. The first segment preserves the chromatic characteristics of the input scene. The second segment
performs chromatic compression in the same way as an equivalent knee function, but at a
slower rate. The third segment performs chromatic clipping out to the \( \text{Max}_{\text{input}} \) point.

\[
output = \begin{cases} 
\text{input}, & 0 \leq \text{input} \leq K \\
\left( \frac{\text{Max}_{\text{output}} - K}{K' - K} \right) \text{input} + K \left( 1 - \frac{\text{Max}_{\text{output}} - K}{K' - K} \right), & K < \text{input} \leq K' \\
\text{Max}_{\text{output}}, & K' < \text{input} \leq \text{Max}_{\text{input}}
\end{cases}
\]  

(1-13)

Recently, Hung-Shing, et al. (1999) have utilized a non-linear scaling function that uses a
power function for compression. The form of this scaling function is given by:

\[
output = \text{Max}_{\text{output}} \left( \frac{\text{input}}{\text{Max}_{\text{input}}} \right)^{\gamma}
\]  

(1-14)

where \(0 < \gamma < 1\). This function has the effect of compressing more as the chroma increases.
Unlike the knee functions described above this function scales all colors within the
gamut. Hung-Shing, et al. (1999) found that \( \gamma \) values between 0.7 and 0.9 worked well
for pictorial images.

1.8.4 Mapping Directions

All of the mapping directions mentioned in the section on clipping, with the exception of
minimum-distance, have been used for chromatic scaling in one way or another (Gentile,
et al. (1990), Pariser (1991), MacDonald (1993), Morovic and Luo (1997a), Montag and
Fairchild (1997)). As with clipping, scaling along these different directions will produce
different effects on the image data. The cusp-point scaling used by Morovic and Luo
has recently been shown to have good general success. Scaling toward this point has been shown to be robust. This was most likely due to the fact that the scaling was performed toward a region of the gamut that has the most chroma.

One technique that has been tried is scaling/clipping along lines of constant saturation, that is, lines that preserve the chroma/lightness relationship (Wolski, et al. (1994), Montag and Fairchild (1997)). A problem with mapping along this direction is shown in Figure 1-13. Constant-saturation scaling or clipping cannot be performed for the points in the shaded region, such as point B, because they never intersect the gamut. Therefore, if constant saturation mapping is performed (clipping or scaling) it can only be done for points to the left of line OC above the cusp-point lightness, in the non-shaded area.
1.8.5 Device- and Image-Dependent Scaling

After the scaling function and the scaling direction are selected, scaling parameters need to be defined. These scaling parameters insure that all of the input color data are mapped into destination gamut before the color values are converted to device coordinates. In general, two methods have been used to set these scaling parameters: device-dependent and image-dependent methods.
The device-dependent scalings are the most general form of gamut-mapping algorithms. By setting the scaling parameters based on the full range of the source gamut, all possible input colors from that device are accounted for. Thus, for a given combination of source and destination device, a general transformation is defined for all possible input images. If a given output device is able to take inputs from multiple source gamuts, then a specific transformation needs to be defined for each input device. In order to complete the scaling processes outlined in Section 1.8.3, the $\text{Max}_{\text{input}}$ and the $\text{Max}_{\text{output}}$ values need to be defined for every region of color space.

For example, consider mapping the point, P, from the source gamut, shown Figure 1-14, into the destination gamut using a device-dependent scaling function. This example shown in Figure 1-14 uses cusp-point scaling. In order to insure that all input points along line CS are within the destination gamut after scaling, the device-gamut approach sets $\text{Max}_{\text{input}}=S$ and $\text{Max}_{\text{output}}=D$. Thus, any point on range of [C S] ends up within the range of [C D] after the compression. This transformation has a maximum compression rate defined by the ratio of D to S (i.e., $\text{Rate} = D/S$.) If this gamut-mapping algorithm used linear-chromatic compression, for example, the point (P) would be mapped to point $P' = P \times (D/S)$. 
Figure 1-14. Parameters need to set the scaling functions.

In general, this strategy causes too much compression. For example, if a given input image only has a maximum input value defined by point I, then a maximum compression rate of only $D/I$ is needed to insure that all of the input points along the scaling line CS (or CI in this case) end up within the destination gamut. This leads to a form of image-dependent transformations that use the maximum input image point along a given scaling line to determine the $\text{Max}_{\text{input}}$. In this example, $\text{Max}_{\text{input}} = I$. Following this process insures
that, for a given scaling line, the minimum amount of scaling is performed to bring all of the input points within the destination range [C D]. Thus, the input point P is scaled to an output point \( P'' = P \times (D/I) \). Less compression is required using an image-dependent transformation than a device-dependent transformation.

There are performance costs associated with using image-dependent transformations. A unique transformation needs to be developed for each input scene. In addition, a pre-processing step is required to define the scaling parameters. This process may be very slow depending on the type of information needed to specify the gamut-mapping operation. Since image-dependent transformations cannot be generalized for all possible inputs, they cannot be coded into the general-purpose LUTs used by color-management systems. However, for systems that do not require fixed device profiles, image-dependent transformations are beneficial because they have the potential to preserve more of the chromatic information of the source image.

1.8.6 Image-Gamut Scaling

The image-dependent scaling process detailed in the previous section can have a dramatic impact on the quality of the gamut-mapped reproductions compared to a device-dependent process. Setting the Max_{input} point according to the image-dependent process insures that all mapped points are inside the destination gamut; however, from an image-reproduction standpoint, it may not be the optimal way to specify chromatic-compression
transformations. For example, consider the case where the point I, shown in Figure 1-14, is an outlying point in the distribution of points that cover the range [C I]. Suppose that a histogram of the input values revealed that 95-percent of the data over [C I] were contained within [C D]. If the gamut-mapping processes used linear scaling defined by I/D, then 95-percent of the input data would be unnecessarily compressed simply because 5-percent of the data were outside the destination gamut. A better approach may be to leave the 95-percent of the data that is within the destination gamut as is and clip the 5-percent of the remaining data to the gamut surface.

Defining the image-gamut by some fraction of the total set of image points is advantageous over describing the image gamut by the maximum input for the scaling lines. This theory was tested in the gamut-mapping experiments detailed in Section 5 (Braun and Fairchild (1999b,d)). A detailed description of the image-gamut specification process is given in Appendix B. In general, the image-gamut was specified to be the 95-percent contour of the 2-dimensional cumulative-image histogram in the \([L^*, C_{ab}^*]\) plane as a function of hue angle.

1.9 Regional Mappings

The large variability in past color gamut mapping studies suggests that ideal gamut mapping depends on image content, preservation of perceived hue throughout color
space, and the extent of the gamut mismatch in various regions of color space. For these reasons, it may be important to consider regional-dependent gamut-mapping strategies.

Wolski, et al. (1994) divided gamuts into specialized regions, in which they could “fine-tune” the gamut mappings. They divided the gamut into three regions: 1.) an achromatic cylinder around the L* axis; 2.) an upper-gamut region above the L* value corresponding to the maximum chroma (cusp point) of the gamut; and 3.) the complement of the second region. For each of these regions, they based the gamut mapping on the image pixel statistics and the gamut mismatch in that region. At the top of the gamut, region 2, they applied soft clipping to the lightness channel first. They then clipped the remaining out-of-gamut points to the surface of the destination gamut, preserving chroma. In region 3, they applied a scaling and shifting process to account for lightness differences between the source and destination gamuts. This was followed by mapping data toward the center of the gamut. Region 1 pixels were affected by the global lightness adjustment function. This may be the only study with a detailed use of region-dependent processing for gamut mapping. Their results show promise for regional techniques.

Spaulding, Ellson, and Sullivan (1995) discussed the gamut-mapping algorithm developed by the Eastman Kodak Company. Similar to Wolski, et al., the Kodak process divided the reference color space into regions. These regions were processed using different gamut mapping functions. The low chroma regions of the color space were mapped colorimetrically and the chromatic regions are warped into the destination gamut
such that they filled it. Due to the proprietary nature of this process, details of the warping process were unclear.

Motomura (1997, 1999) defined a categorical gamut-mapping algorithm by specifying like regions of color name between the source and destination gamuts. These data are used to populate his gamut-mapping transformation. This process uses a series of multivariate-statistical transformations to map points from the source gamut to the destination gamut.

Luo and Morovic (1998) suggested the need for regional dependent processing. They gave examples where gamut-mapping lines are drawn differently based on specific gamut differences and regions of color space. Their results showed that their regional dependent algorithms did not perform as well as the simpler process of GCUPS.

Montag and Fairchild's research (1997) showed that different global functions work better depending on whether the gamut mismatch occurs at the top or the bottom of the gamut.

1.10 Gamut Expansion

Gamut expansion can be used in addition to compression in a gamut-mapping strategy for generating preferred reproductions. (In this thesis, gamut expansion will refer to algorithms that include compression and expansion.) For example, suppose the input image comes from a scan of a SWOP print and the output device is a thermal printer. The
SWOP gamut is smaller than that of many thermal printers. Reproducing the SWOP image colorimetrically on the thermal printer is probably only appropriate for matching or proofing tasks. The extra chromatic and lightness dynamic range available in the thermal printer gamut can produce a more pleasing image. Duplication systems, such as copiers and faxes, might also benefit from using gamut-mapping algorithms that include gamut expansion. These systems may have larger output gamuts than the input documents and gamut expansion can lead to enhancement of the reproductions. Gamut expansion might play a role in Internet web viewing (world-wide web) in generating more preferable reproductions. The input may be a scanned photograph or print. The output device, a monitor, has a much greater dynamic range in the high lightness greens, blues, reds, and oranges than a typical reflective sample. Using the extra monitor dynamic range may make a company's product more desirable than its competitor's.

The Kodak UltraColor gamut-morphing process described by Spaulding, Ellson, and Sullivan's (1995) addresses gamut expansion. The device primaries of the input device were mapped toward the corresponding device primaries of the reproduction device. Colors near the device primaries in business graphics are reproduced near the primaries of the output device. This results in higher chroma reproductions, which may be preferred for graphics viewing.
1.11 **Limitations of Current Algorithms**

Current gamut-mapping algorithms described in the literature focus on the shapes of the device gamuts and not the image data. Contradictions in the results of these studies may result from establishing the mapping strategies on select sets of images. Attention needs to be focused on maintaining the image attributes of the scene such as lightness and chromatic contrast to generate gamut-mapped images that maintain the appearance attributes of the original images. Some specific limitations of state-of-the-art techniques are described in the following sections.

1.11.1 Color Space

Until recently, very little has been published about the implications of performing color gamut mapping in a color space that has non-linear perceived hue lines (Hung and Berns (1995), Ebner and Fairchild (1998a)). The negative consequences of performing chroma compression while preserving metric hue angle in a color space such as CIELAB have become very apparent (Braun, Ebner, and Fairchild (1998)). When a color is compressed along a line of constant metric hue angle in CIELAB, a perceived hue shift may result depending on the hue region of interest and the difference in lightness or chroma between the original and the reproduced color. Studies by Marcu (1998) and McCann (1999) were based on the Munsell data and, as such, may not accurately correct regions of high chroma that are present on a monitor.
In order to overcome the hue linearity limitations of the CIELAB color space, a hue-linearized version of CIELAB was generated using corrective LUTs based on the Hung and Berns (1995) visual data. This is discussed in detail in Section 3. These LUTs were tested versus those generated from Ebner and Fairchild (1998a) data. The results of these experiments showed that the Hung and Berns data set provides better hue correction than the Ebner and Fairchild data set in the “blue” region of color space. The remainder of the CIELAB color space produced as uniform or more uniform results than either of the hue-corrected spaces tested. As a result, a hybrid color space was generated using CIELAB as the base and applying the Hung and Berns hue-correction data only in the “blue” region of color space. Therefore, a sound color space has been generated for gamut mapping purposes that solves the purple/blue color shifts generated using CIELAB. This space can also be used in the future as a benchmark for testing uniform-color spaces such as Munsell and the IPT space generated by Ebner and Fairchild (1999b) for their utility in color gamut mapping.

1.11.2 Lightness Rescaling

In general, linear lightness rescaling is still widely used in the gamut mapping literature. Morovic and Luo (1997a,b, 1998) and Montag and Fairchild (1997) used linear lightness scaling as the first stage in their gamut mapping processes. The philosophy has been that this type of scaling preserves the relative positions of the image pixels in the new
lightness dynamic range. This process immediately results in an overall lightening of the image and globally reduces the contrast of the image features.

Other lightness rescaling strategies use soft-clipping, transitive (shifting) (Wolski, et al (1994)), or clipping to account for the differences in dynamic range between the source and destination devices. These rescaling techniques, except for clipping, result in remapped images that have lower perceived contrast and are lighter than the original image. Clipping preserves the lightness of much of the image but can result in loss of detail in low-lightness features.

Lightness rescaling must be performed in such a manner that it overcomes the apparent loss in visual contrast that results when the image is scaled into a smaller dynamic range. In addition, the lightness rescaling has to be tailored so that a major portion of the output-dynamic range is reserved for the lightness features that are most important to the scene. The proposed approach utilizes a series of optimized sigmoidal-lightness remapping functions (Braun and Fairchild 1999a,c). Details of the sigmoidal mapping are given in Section 4. The form of these functions is image-dependent and can be determined from simple lightness histogram statistics about the image. In addition, these sigmoidal lightness-remapping functions are tailored to the dynamic-range differences between the devices, as well as the lightness-key of the image.
1.11.3 Chroma Mapping

Studies performed by Montag and Fairchild (1997) and Gentile, et al. (1990) suggest there are limitations to using linear chromatic-compression functions. In general, linear chromatic-compression desaturates the entire scene. Based on the nature of gamut mapping, this does not have to happen. Some of the input image points are naturally within the destination gamut. As such, care needs to be taken before these points are arbitrarily compressed. The knee functions and the soft-compression used throughout the literature were specifically designed to leave the low-chroma in-gamut pixels unchanged for the reasons discussed above. Therefore, whenever chromatic compression is required, these functions need to be considered first.

Chromatic clipping alleviates the problem of desaturating low-chroma in-gamut colors, but, as was shown is Section 1.8.2, the possibility exists for several serious artifacts. As such, it is useful to take the spirit of clipping algorithms (i.e., only compress high-chroma colors that are out of gamut) and apply it to a scaling function. In this thesis, clipping was replaced by a knee function with a knee point set to 90-percent of the destination gamut range. This function leaves most of the in-gamut-input pixels unchanged and only compresses in-gamut pixels near the edges of the gamut. In addition, instead of just clipping the out-of-gamut pixels to the destination surface 10-percent of the gamut is reserved for these features. This helps to reduce the clipping artifacts previously discussed.
1.11.4 Gamut Expansion

Very little has been done to generalize a solution for gamut expansion of pictorial images. The solution proposed by Spaulding, Ellson, and Sullivan (1995) is for multi-purpose printing systems that need to perform well for both business graphics and pictorial scenes. Their process offers a trade-off when mapping business graphics and pictorial images, sacrificing hue and lightness for chroma. The lightness difference between a CRT blue and an ink-jet blue can be as much as 50 $L^*$ units. In addition, by warping the CRT primaries onto the printer primaries, the hue of the colors is shifted. The hue and lightness shifts will cause the monitor blue to shift to dark purple on most printers. This is probably not optimal for pictorial images.

In this thesis, a general gamut-expansion process is offered for pictorial images that is based on the inverse gamut-compression algorithms that performed well for gamut-compression.

1.12 Gamut-Mapping Experiments

Based on the limitations mentioned above, a series of psychophysical gamut-mapping experiments were performed to test the following hypotheses about gamut mapping for pictorial images.

1.) Gamut-mapping algorithms that operate in a hue-linearized color space such as Hung and Berns hue-corrected CIELAB (Braun, Ebner, and
Fairchild (1998)) will outperform the same algorithms processed in a color space with non-uniform hue lines, such as CIELAB. (Section 3).

2.) Contrast-preserving lightness-scaling algorithms will perform better than those algorithms that perform linear lightness scaling (Braun and Fairchild (1999a,c)). (Section 4 and 5)

3.) Applying an image-gamut approach to chromatic scaling will perform better than device-gamut approaches (Braun and Fairchild (1999b)). (Section 5)

4.) Chromatic compression using a knee function that acts very similar to chromatic clipping will perform better than linear scaling (Braun and Fairchild (1999b,d)). (Section 5)

5.) A gamut-expansion process that functions as the inverse of the compression function outlined in items (2) – (4) above will produce reproductions that are preferred to gamut-mapping functions that only perform compression. In addition, this gamut-expansion process will perform better than a process that is based on inverting the linear lightness and chromatic compression algorithms. (Section 6)
2 Techniques for Gamut Surface Definition and Visualization

2.1 Summary

Accurate techniques for defining and visualizing a device’s gamut boundary are very important in the design of robust color gamut mapping algorithms. A novel technique for defining the surface of a color imaging device’s gamut in CIELAB L\(^*\)C\(_{ab}\)\(^*\)h\(_{ab}\) color space using a triangulation and interpolation process is presented. This process provides an accurate approach for gamut surface fitting, from measured or modeled data, that is independent of gamut concavity or convexity. The results of a goodness-of-fit test indicate that the gamut surface can be predicted to a mean \(\Delta E_{ab}^*\) of 1.1, for the monitor gamut tested. In addition, the L\(^*\)C\(_{ab}\)\(^*\)h\(_{ab}\) space is shown to be useful for several gamut mapping and visualization tasks.

2.2 Introduction

Color gamut mapping is an integral part of color management. It is important to be able to accurately model the gamut surfaces for all of the devices in a color image reproduction chain. Often in a color reproduction chain, the source and destination gamuts are dissimilar. Therefore, in order to obtain high-quality color reproductions between these devices, some type of color gamut mapping must occur. In order to perform gamut mapping in a visually effective manner, a description of a device’s color
gamut must be obtained. Several authors have given processes for color gamut specification (McBride (1996), Herzog (1996), Inui (1984), Balasubramanian and Dulal (1997)).

The process described in this thesis defines the gamut boundaries of color imaging devices that is useful for visualizing the gamut volume in two and three dimensions. The gamut surface estimation is based on well-established surface-fitting procedures for generating a uniform grid of points based on a set of non-uniformly spaced input points. These techniques are commonly used by mathematical analysis software packages such as MATLAB™ and IDL™. The approach presented here applies these relatively straightforward processes to simplify the estimation of an imaging device’s color gamut.

The following sections will provide: 1.) a description of the triangulation and interpolation process used to convert non-uniformly spaced gamut-surface data, derived from a device RGB cube, into a CIELAB L’C_ab’h_ab (mountain-range) representation of the color gamut; 2.) a "goodness-of-fit" test that was used to gauge the accuracy of the gamut surface estimation process; and 3.) the benefits of using the mountain-range representation for gamut mapping and gamut visualization.

2.3 Gamuts with Known Connectivity

In order to perform color gamut mapping, image data should be in a reference color space that is visually based. For gamut mapping this requires transformation of both image data
and device color gamuts into a reference visual color space. The device-dependent 
representation of a three-color imaging device's gamut can be generalized by its 
RGB/CMY colorant cube. The surface of the RGB/CMY cube can be thought of as all 
one- and two-primary mixtures as well as all three-color mixtures when at least one of the 
primaries is at a maximum. The eight corners of this cube correspond to device red, 
green, blue, cyan, magenta, yellow, black, and white. The device-independent (e.g., CIE 
XYZ or CIELAB) representation of a color imaging device's gamut is a nonlinear 
transformation of its device-dependent representation. Calculation of the gamut surface 
consists of transforming the RGB/CMY cube surface into corresponding values in the 
reference color space, either using a physical model or printing and 
spectrophotometrically measuring selected values from the surface of the RGB/CMY 
cube. This process can be generalized to most four-color printing systems by processing 
the RGB/CMY cube through the appropriate transformation to CMYK. The gamut 
boundary can be obtained by connecting the gamut surface points in CIELAB using the 
same connectivity that these points had in the RGB/CMY cube.

2.4 Gamuts with Unknown Connectivity

The process for connecting the surface points for a device gamut is much more difficult 
when the inherent connectivity of the points is not known from the colorant cube. This 
may be the case when defining the gamut from a set of points from a Q60 target or a 
similar data set. Once these data are in the desired reference color space, computer
graphics algorithms such as convex-hull routines can be used to form a polygon mesh encompassing the data (Preparata and Shamos (1995)). This process only works well if the gamut surface is convex. The convex hulling process will mask concavities on the surface. In order to generate an accurate representation of these gamuts, Balasubramanian and Dalal (1997) defined a process that transformed the gamut surface data such that it could be represented by a truly convex set. The gamut-surface points of the convex set were connected using a convex-hulling process. This technique produces the desired connectivity list as long as a monotonic transformation could be found to generate the convex set. When applied to the original CIELAB data, an accurate gamut hull was obtained.

2.5 CIELAB $L^*C_{ab}^*h_{ab}$ Gamut Specification

In order to make the gamut mapping process more efficient it would be useful to have the gamut represented in a form that allows easy access to the key components of the gamut. Some of these components are:

- a line-gamut boundary as a function of hue angle. (Note: The line-gamut boundary is represented by plotting gamut-surface lightness as a function of chroma at a single hue angle, Figure 2-1),
- the cusp point as a function of hue angle. (Note: The cusp point is the achromatic point that has the same lightness as the point of maximal chroma for a given line-gamut boundary, Figure 2-1), and

- the chroma of the gamut boundary at any \([L^*, h_{ab}]\) coordinate. (Note: This is useful for identifying out-of-gamut pixels.)

![Diagram of line-gamut boundary and cusp point](image)

*Figure 2-1. Illustration of a line-gamut boundary plot and the gamut cusp point for a given hue angle in a reference color space such as CIELAB.*

As will be shown in Section 2.9, some of these gamut-mapping tasks are more efficiently performed when a device's gamut is represented in cylindrical coordinates, CIELAB
L*Cab*h_ab. In this form, the C_ab* component of the gamut surface is functionally related to L* and h_ab. Therefore, for any hue angle, each lightness is represented by a single chroma value. This functional relationship between the lightness and chroma of the gamut surface is an essential part of describing the gamut surface in CIELAB L*C_ab*h_ab coordinates.

Herzog (1996) defined an analytical approach to relate the chroma of the gamut surface to lightness and hue angle. While accurate, this process is very mathematically complex. In order to simplify this process, a gridding and interpolation process was used to generate the gamut structure in CIELAB L*C_ab*h_ab space.

2.5.1 Triangulation and Interpolation

Uniformly spaced points on a device RGB cube will typically be non-uniformly spaced in CIELAB due to the nonlinear relationship between the two spaces. When these data are converted into L*C_ab*h_ab and projected into the L*h_ab plane they lie on an irregular grid. In order to convert this irregular or non-uniform set of points into a regular grid, interpolation is needed. One such interpolation process involves using triangular interpolation among the data points in the L*h_ab plane.

The current technique transforms the gamut surface data into cylindrical CIELAB coordinates (i.e., L*C_ab*h_ab) and performs a triangulation and gridding process to specify the surface of the gamut. The triangulation of the data is performed by projecting the nonlinearly spaced L*C_ab*h_ab data from the device RGB cube onto the L*h_ab plane. The
data in the L*h* plane are grouped into triangles using the inherent connectivity of the points in the RGB cube, Figure 2-2. The vertices of this mesh represent either measured or modeled data from the surface of the RGB cube. In order to insure that the line-gamut boundary at h*=0° and h*=360° is the same, the original data are periodically replicated on either side of the 0-360° range, Figure 2-3. Using the triangle list formed from the RGB cube connectivity and the corresponding C* for each triangle vertex, a uniform grid of C* values is interpolated over the L*h* plane using triangular-linear interpolation, Figure 2-4.

The interpolation process is performed using the barycentric weights formed by the sub-triangles shown in Figure 2-4. The estimated chroma for any point within the triangular polygon is given by:

\[ C^*(u, v, w) = u \cdot C_u^* + v \cdot C_v^* + w \cdot C_w^*, \quad (2-1) \]

\[ u = \frac{A_u}{A}, \quad (2-2) \]

\[ v = \frac{A_v}{A}, \quad (2-3) \]

\[ w = \frac{A_w}{A}, \quad (2-4) \]

where \( A_u, A_v, \) and \( A_w \) are the areas of the sub-triangles and \( A \) is the area of the triangle formed by \( C_u^*, C_v^*, \) and \( C_w^*. \) In order to use this process to build a regular grid of points in the L*h* plane, two algorithms are needed. The first is a searching algorithm that finds
which triangular polygon contains a given \([L^*, h_{ab}]\) point. The second algorithm calculates the interpolated value. For this thesis the TRIANGULATE and the TRIGRID functions built into IDL® were used to perform these calculations. The IDL® programs used to calculate the mountain-range gamuts are given in Appendix C.

![Diagram](image)

*Figure 2-2. Projection and triangulation of uniform RGB cube vertices into \(L^*h_{ab}\) plane. Each of the vertices has a corresponding \(C_{ab}^*\) value.*
Figure 2-3. Periodic replication of the measured RGB cube data to insure connectivity between $h_{ab}=0^\circ$ and $h_{ab}=360^\circ$.

Figure 2-4. Illustration of triangular linear interpolation. The interpolated value is calculated by weighting the chroma of each vertex by their adjacent area fractions given $u=A_v/A$, $v=A_w/A$, and $w=A_u/A$. 

56
The uniform grid points are interpolated for steps of $\Delta h_{ab} = 1^\circ$ and for $\Delta L^* = 1$. The resulting uniform $C_{ab}^*$ grid in the $L^*h_{ab}$ plane is represented by a 101x361 element matrix. The chroma, $C_{ab}^*$, of the gamut surface for any $[L^*,h_{ab}]$ coordinate can be estimated from the uniform $C_{ab}^*$ grid using bilinear or cubic-convolution interpolation procedures. The following series of figures gives different representations of the uniform grid of $C_{ab}^*$ values for a typical monitor with no external or internal flare terms present (i.e., device black can achieve an $L^*=0$). In Figure 2-5, the intensity of each pixel is proportional to the chroma (i.e., low/high intensity corresponds to low/high chroma). The surface plot shown in Figure 2-6 is referred to as a mountain-range gamut based on the peak-like structure of the gamut surface in cylindrical CIELAB coordinates. The height of the surface from the $L^*h_{ab}$ plane is $C_{ab}^*$.

The main assumption in this process is that the chroma of the black point and white point drop off to zero. This is necessary since in the $L^*C_{ab}^*h_{ab}$ representation the black point and white point are replicated for all hue angles at the $L^*$ for which they occur. Therefore, the $C_{ab}^*$ for these two points are forced to zero if they are not already zero. In general, this assumption has held approximately true for the devices examined thus far.
Figure 2-5. $L^*C_{ab}^*h_{ab}$ representation of a CRT gamut generated using triangulation and interpolation. Vertical scale represents $L^*$ and horizontal scale represents $h_{ab}=[0^\circ,360^\circ]$.

Figure 2-6. Mountain-range representation of the $L^*C_{ab}^*h_{ab}$ CRT gamut generated from triangulation and interpolation.
2.5.2 Line-Gamut Boundary Specification

A useful feature of the $L^*C^{ab}_*h^{ab}$ gamut-surface representation is that individual hue angle slices can be quickly extracted from the $C^{ab}_*$ matrix and visualized in a 2D plot, Figure 2-7. These slices prove to be very useful in designing color gamut mapping algorithms that are customized on a hue-angle dependent basis. If the gamut data were represented as a 3D wire-frame mesh in CIELAB space, an estimation process would be required to extract a slice profile of the gamut surface for a given hue angle. This might involve a series of ray-tracing steps where the gamut intersection points would be located for a series of $L^*$ values for the given hue angle. Such a process is more computationally demanding than looking-up or interpolating values from a pre-computed 2D matrix.

![Figure 2-7](image_url)

*Figure 2-7. Illustration of a slice taken from the mountain-range representation of CRT gamut.*
2.6 CIELAB Volume Representation

It may be desirable to transform the uniform L*C*a*b* grid into rectangular coordinates, L*a*b*, Figure 2-8. This will generate a highly faceted wire-frame model of the triangulated and interpolated gamut. Neighboring points in the L*C*a*b* representation remain neighboring points in CIELAB. Therefore, the connectivity of a polygon mesh in L*C*a*b* space is the same in CIELAB representation. As such, no hulling procedure is required to produce a polygon mesh in CIELAB. This polygon mesh can then be used for gamut mapping, for gamut mismatch visualization, and for visualization of image pixel data within the source and destination gamuts. All these processes aid in the development of color gamut mapping algorithms.

Figure 2-8. Typical CIELAB "wire-frame" representation of the triangulated and interpolated mountain-range gamut of a monitor. (Note: There is a smooth seam located along the a* axis where the gamut is connected between 0° and 360°.)
2.7 Noise Filtering and Smoothing

The CIELAB $L^*C_{ab}^*h_{ab}$ gridding process for gamut-surface estimation offers a straightforward technique for reducing the effects of measurement noise and print-to-print variability in the measured RGB data used to generate the gamut surface. The mountain-range gamut that results from the triangulation and interpolation process is represented by a matrix. Noise in the data used to generate the mountain-range will cause the gamut surface to not be smooth, Figure 2-9. To illustrate this point, white noise generated from a uniform random-number generator was added to the chroma of the vertices of the gamut surface. The gamut used in this example came from a gain-offset-gamut model (GOG) for a monitor (Berns, Motta, and Gorzynski (1993)). After interpolation, the mountain-range gamut was convolved with low-pass filter. This filtering reduced the noise texture in the gamut surface, Figure 2-10. In this case, a 5x5 unity-gain (i.e., the filter elements sum to one) boxcar-averaging filter was used. Other filter functions might be optimal depending on the structure of the surface noise. Minimally, operations such as median filtering can be applied for this purpose.
Figure 2-9. Mountain-range gamut of a monitor with simulated measurement and print-to-print variability. White noise was added to the chroma of the vertices prior to triangulation and interpolation.

Figure 2-10. Noise-filtered mountain-range gamut of the gamut shown in Figure 2-9.
2.8 Error Analysis

The objective of this test was to determine the accuracy of specifying the gamut surface of a monitor gamut using the interpolation and gridding process discussed in Section 2.5.1. The data used to specify the gamut surface came from a 20x20 RGB grid sampling of each gamut face. These data were converted to CIELAB using a GOG model for the monitor (Berns, Motta, and Gorzynski, 1993).

2.8.1 Procedure

The CIELAB data for the monitor gamut were generated using a GOG model with no flare terms present. The CRT mountain-range gamut was generated using the following procedure:

1. Generate 20x20 nonlinear RGB grid of points for each of the six faces of the RGB cube. The spacing of the grid points on the faces of the RGB cube was more near the edges and corners than in the center.

2. Convert the RGB digital counts to CIELAB using the GOG monitor model.

3. Convert CIELAB data to CIELAB $L^*C_{ab}^*h_{ab}$. 
4. Generate a uniform grid in the \([L^*, h_{ab}]\) plane of \(C_{ab}^*\) values using the triangular interpolation process. The grid spacing was \(\Delta h_{ab} = 1^\circ\) and \(\Delta L^* = 1\). A 3x3 boxcar-averaging filter was applied to reduce the effects of aliasing. The convolution was not performed with points at the edges of the \(C_{ab}^*\) image to avoid wrap around effects of circular convolution.

Once the mountain-range gamut was defined, a random sampling of 6000 surface points from an RGB cube was generated. These points were converted to CIELAB \(L^*C_{ab}^*h_{ab}\) values using the GOG model. For each of the 6000 points, a chroma value was interpolated from the \(C_{ab}^*\) mountain-range gamut at the \([L^*, h_{ab}]\) values corresponding to the modeled value. The modeled CIELAB values from the GOG model were then compared to the estimated CIELAB values derived from the mountain-range. (Note: The difference between modeled CIELAB and estimated CIELAB is that the chroma value for the estimated CIELAB were derived from the mountain-range gamut).

### 2.8.2 Results and Discussion

The values given in Table 1 show that there was little difference between the modeled-gamut, predicted from the GOG device model for the monitor, and the mountain-range estimated gamut. A plot of the histogram of \(\Delta E_{ab}^*\) errors for all 6000 points confirms that the fit between the estimated mountain-range gamut and the device model are accurate, Figure 2-11.
Table 1: Results of error analysis. Comparison of mountain-range predicted gamut surface to gamut surface predicted by CRT GOG model

<table>
<thead>
<tr>
<th>Number of Points used in Analysis:</th>
<th>6000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean $\Delta E_{ab^*}$:</td>
<td>1.06</td>
</tr>
<tr>
<td>Variance in $\Delta E_{ab^*}$:</td>
<td>3.74</td>
</tr>
<tr>
<td>Max $\Delta E_{ab^*}$:</td>
<td>30.15</td>
</tr>
<tr>
<td>No. of Points with $\Delta E_{ab^*} &gt; 5.0$:</td>
<td>126</td>
</tr>
</tbody>
</table>

Figure 2-11. Histogram of $\Delta E_{ab^*}$ errors between GOG modeled points and points estimated from the mountain-range gamut. (Note: The histogram is plotted on a logarithmic axis so that the high $\Delta E_{ab^*}$ errors can be identified.)

There are a few points about the results to discuss. The first is the relatively large maximum $\Delta E_{ab^*}$ that resulted from the analysis. The graphs given in Figure 2-12 represent vector error plots for all of the test points that resulted in $\Delta E_{ab^*}$ values greater than 5.0 (126 points out of 6000). Figure 2-12a shows that the majority of the errors are
in estimated chroma (i.e., the estimated chroma is lower than the modeled chroma). Nearly all of these errors occur in about the same hue angle region (i.e., approximately between an \( h_{ab} \) of 90° and 130°). These hue angles line up well with the yellow CRT secondary. Figure 2-12b shows that the majority of large errors are in the estimation of \( b^* \). The \([C_{ab}^*, L^*] \) plot indicates that the large errors occur for the high lightness samples (i.e., \( L^* > 80 \)).

The reason for these large errors is that, in forming the \( C_{ab}^* \) grid (mountain-range), the location of the yellow primary is shifted slightly in hue angle as a result of the discrete location of the uniform grid points. Since the peak for the yellow primary is very steep coming down from "white" (\( L^* = 100 \)), any misalignment in the uniform interpolation grid may cause large errors in the estimated chroma. The effect at other peaks is not as pronounced as that for the yellow primary since the data at these peaks do not vary as rapidly as the data near the yellow peak. (The yellow peak increases about 120 \( C_{ab}^* \) units over only a \( \Delta L^* \) region of about 6 units and a \( \Delta h_{ab} \) region of about 2°.) Based on the grid spacing, the shift would be on the order of 0.5° in \( h_{ab} \) and up to 0.5 units of \( L^* \). It is the position of the "yellow peak" and not its \( C_{ab}^* \) value that is in error. Therefore, this phenomenon should not significantly effect the results of any gamut mapping experiments or the appearance of gamut mapped images.
Figure 2-12. CIELAB error vector plots for sample points whose $\Delta E_{ab^*}$ was greater than 5.0. (The arrows point from the modeled value to the estimated value.)
2.9 Advantages of "Mountain-range" Gamuts for Gamut Mapping

2.9.1 Gamut Specification

The gridding and interpolation process in the \( L^*C_{ab}^*h_{ab} \) color space has several distinct advantages over specification in CIELAB using convex hull algorithms. First, data concavity is not an issue. The surface fitting process will work equally well on concave data sets as convex data sets. Additionally, it is easy to determine how interpolated surface points are connected to other points in the lattice based on their position in the \( L^*h_{ab} \) plane.

Device models are not necessary to generate the gamut surface of a device; the gamut can be estimated from measured data directly. This process proves to be a robust surface-fitting algorithm in the presence of measurement noise or print-to-print variability. Measured data is inherently noisy. This noise will cause micro-concavities in the surface structure of the gamut. Convex hull routines always over-predict the surface of the gamut by masking the micro-concavities. The process described here will fit this surface texture automatically. Also, the ability exists to filter out unwanted noise through straightforward image processing.
2.9.2 Gamut Mismatch Estimation

Gamut mismatch can be quickly evaluated using line-gamut boundary plots of the two gamuts. This is accomplished by selecting the same hue-angle column from the two mountain-range gamuts and plotting them in a two-dimensional lightness versus chroma plot, Figure 2-13. This representation is an exceptionally intuitive gamut difference visualization technique but only gives information about the gamut differences for the given hue angle.

![Figure 2-13. Comparison of two gamuts using a line-gamut boundary representation.](image)
For a more global gamut comparison, the difference between two mountain-range gamuts can be taken. The resulting matrix contains both the direction (positive or negative difference) and the magnitude of the gamut mismatch in $C_{ab}^*$ units. This makes it easy to locate regions where the source gamut is larger or smaller than the destination gamut. This type of information is key to performing gamut mapping. The difference gamut can be represented as either a surface plot or a contour plot as shown in Figure 2-14 and Figure 2-15. The surface plot gives a visual estimation of the gamut differences. While more difficult to visualize, the contour plot gives a more quantitative representation of the gamut differences.

Figure 2-14. Differences between a monitor gamut and a thermal printer gamut shown as a surface plot.
The benefits of the gamut difference process extend to evaluation of individual image pixels as well as the source and destination gamuts. To determine if a given source image pixel is within or outside of the destination gamut, the $C_{ab}^*$ value of the destination gamut surface is estimated by interpolating the $[L^*, h_{ab}]$ coordinate of the source pixel. The $C_{ab}^*$ value of the source pixel is then compared to that of the destination gamut. If this difference is negative, the source pixel is outside of the destination gamut. If the difference is positive, the source pixel is within the gamut of the destination device.
These types of processes using a CIELAB wire-frame representation of the gamut are more computationally complex calculations and ray-tracing algorithms are often required.

2.9.3 Gamut Mapping

As shown in Section 2.5.2, it is very easy to get to the line-gamut boundary using the mountain-range gamut representation. The fact that the gamut boundary lines are pre-calculated makes the gamut-mapping process proceed much more quickly than if these boundaries were calculated as the image data were mapped.

The gamut-mapping algorithms given in Sections 5 and 6, of this thesis, rely on identifying the cusp point of the gamut as a function of hue angle. This process is greatly simplified using the mountain-range gamut representation. For a given hue angle, the cusp point is obtained by storing the line-gamut boundary vector in a temporary vector and determining the index that corresponds to the maximum value. This index is the lightness of the cusp point. In order to perform this calculation from a CIELAB wire-frame gamut, many more computations would be required.

2.10 Conclusions

The results of the analysis presented in Section 2.8 indicate that a CIELAB L*C_{sb}*h_{sb} gamut can be generated that accurately represents the surface structure of a color imaging device's gamut. This gamut fitting process allows the generation of an imaging device's
color gamut without the necessity of a device model. The triangulation and interpolation process presented will operate regardless of whether the source data is convex or concave in nature. The ability to filter out measurement noise and sample variability is a powerful feature of this technique. Finally, the gamut data is in a form that is very computationally efficient for many gamut-mapping tasks.
3 Color Gamut Mapping in a Hue-Linearized CIELAB Color Space

3.1 Summary

It is desirable to preserve the perceived hue of the original scene when color gamut mapping pictorial images for matching tasks. A distinction is made between perceived hue, the visual sensation of hue, and metric hue angle (e.g., CIELAB hue angle ($h_{ab}$).) If a gamut-mapping task constrains CIELAB metric hue angle in the "blue" region of CIELAB, a perceived-hue shift will result. Due to these non-linearities, two hue-linearized versions of the CIELAB color space were generated, one from the Hung and Berns (1995) visual data and one from the Ebner and Fairchild (1998a) data set. These modified versions of the CIELAB color space were psychophysically tested for their hue-linearity characteristics against the CIELAB color space. The results of these experiments showed that, in the "blue" region of CIELAB (i.e., $260^\circ < h_{ab} < 320^\circ$), the hue-corrected color spaces were more visually uniform and perform better than CIELAB in gamut mapping. However, outside of the blue region, the CIELAB color space performed as good as, or better than, either hue-corrected space. As such, a modified CIELAB color space was developed that only performed the hue correction in the “blue” region of color space. This process eliminates the “purple” shift that results when mapping “blues” in CIELAB.
3.2 Generation of Hue-Correcting Look-Up Tables

In the development and testing of a hue-corrected color space, two data sets were used to generate color-space transformation look-up tables. The first was the Hung and Berns data set (1995), referred to throughout as the Hung and Berns data. The second set of data and corresponding transformations were developed by Ebner as part of his Ph.D. dissertation (Ebner (1998), Ebner and Fairchild (1999a)). The data used in his process are referred to as the Ebner and Fairchild data throughout. Sections 3.2.1 and 3.2.2 detail the processes used to generate these two sets of transformations.

3.2.1 Hung and Berns Data Set

The Hung and Berns data set consisted of 12 lines of constant perceived hue that uniformly spanned the CIELAB hue circle, Figure 1-3. Hung and Berns (1995) performed an adjustment experiment using a monitor to create their constant hue data. This is significant because they were able to generate visual data for very high chroma levels where the CIELAB color space has many of its biggest hue non-linearities. For a given hue angle, each hue line consisted of 4 points at constant lightness and constant perceived hue. The points varied in chroma by 25, 50, 75, and 100 percent of the maximum chroma that the monitor gamut would allow. In order to get the maximum range of possible chroma, these hue lines occurred at the lightness levels of the monitor primaries and secondaries.
These data were used to create a two-dimensional LUT, in which the linearity of the perceived hue lines was assumed to be invariant with lightness. This assumption was predicated on the fact that the Hung and Berns data set did not contain complete lines of constant hue at multiple lightness levels. It was also believed that any lightness dependency was secondary to the effect of correcting the 2-D hue errors of CIELAB.

The Hung and Berns data set were converted to a LUT using the following sequential linear interpolation process (Note: The Matlab™ code that performed these calculations is given in Appendix D):

1. Complete the Hung and Berns data set so that each of the 12 hue lines were defined from the neutral axis \([a^*=0, b^*=0]\) out to \(C_{ab^*} = 150\). Extrapolation of the data back to the neutral axis consisted of a simple linear fit between the first point in the data series to the neutral axis. Extrapolation out to \(C_{ab^*} = 150\) was accomplished by determining the intersection point between the line formed by the last 2 points in the data series and a chroma circle at a radius of 150 chroma units, Figure 3-1. Thus, the last three data points shown in a given hue line in Figure 3-1 are colinear.

2. Convert the complete data set to CIELAB \(L^*C_{ab^*}h_{ab}\) coordinates.
3. Determine the base hue angle for each of the 12 hue lines. Base hue angle was defined by the CIELAB hue angle of the first Hung and Berns point in the each hue-line series. These base hue angles are shown in Appendix E along with the hue and chroma data for each of the 12 hue angles.

4. Generate a gridline for each of the 12 base hue angles by linearly interpolating the CIELAB-hue angle, as a function of $C_{ab}^*$, from the base hue angle. This is accomplished using the following algorithm:

```plaintext
let,

gridLines = 12 x 151 matrix to store the gridlines
H&BCchromaData = 12 x 6 matrix of chroma values from the Hung and Berns visual data.
H&BhueData = 12 x 6 matrix of hue values from the Hung and Berns visual data.

For each gridline from 1 to 12

  currentHueLineChroma = H&BCchromaData(gridline, all columns)
  currentHueLineHue = H&BhueData(gridline, all columns)

For each $C_{ab}^*$ from 0 to 150

  gridLines(gridline, $C_{ab}^*$) = interp(currentHueLineChroma,
                                            currentHueLineHue,
                                            $C_{ab}^*$)

Next $C_{ab}^*$

Next gridline
```
where \textit{currentHueLineChroma} and \textit{currentHueLineHue} are six element vectors containing the Hung and Berns visual data for the current hue line being processed. The function \textit{interp} performs linear interpolation between the input points. The output from this algorithm is a 12 x 151-element matrix of the gridlines. The 12 Hung and Berns gridlines are shown as the solid lines in Figure 3-2.

5. Generate a complete set of gridpoints (i.e., filling in the LUTs) by linearly interpolating between the 12 gridlines at fixed chroma intervals on the range of \{0,150\}, every one-degree in hue angle from \{0,360\}. Depending on the direction of the interpolation, between base hue angle and CIELAB-hue angle, the direction of the LUT transformation is controlled. This algorithm is given by the following:
let,

\( \text{baseHue} = \{ \text{baseHue}(1), \text{baseHue}(2), \ldots, \text{baseHue}(12) \} \)

\( \text{inverseLUT} = 151 \times 361 \) element matrix

\( \text{forwardLUT} = 151 \times 361 \) element matrix

**For each** \( C_{ab}^* \) **from 0 to 150**

**For each** gridline from 1 to 12

\( \text{cielabHue(gridline)} = \text{gridLines(gridline, } C_{ab}^*) \)

**Next** gridLine

**For each** \( h_{ab} \) **from 0 to 360**

\( \text{inverseLUT}(C_{ab}^*, h_{ab}) = \text{interp(baseHue, cielabHue, } h_{ab}) \)

\( \text{forwardLUT}(C_{ab}^*, h_{ab}) = \text{interp(cielabHue, baseHue, } h_{ab}) \)

**Next** \( h_{ab} \)

**Next** \( C_{ab}^* \)

where \( \text{baseHue} \) is a 12-element vector of the base hue angles, \( \text{inverseLUT} \) is the LUT transformation from hue-corrected CIELAB to CIELAB, and \( \text{forwardLUT} \) is the transformation from CIELAB to hue-corrected CIELAB. The forward and inverse LUTs are visualized in Figure 3-3 and Figure 3-4.
Figure 3-1. Extrapolation of Hung and Berns data out to $C_{ab^*} = 150$. 
Figure 3-2. Twelve gridlines generated from the Hung and Berns data set. These gridlines are used to populate the 2D LUT.
Figure 3-3. Samplings from the forward hue-correction transformation every 8 degrees in hue angle. Defines the transformation from CIELAB to hue-corrected CIELAB.
Figure 3-4. Samplings from the Inverse hue-correction transformation every 8 degrees. Defines the transformation from hue-corrected CIELAB to CIELAB.

3.2.2 Ebner and Fairchild Data Set

Fritz Ebner supplied the hue-correction transformations that were constructed from the Ebner and Fairchild data set. Details of the creation of these LUTs can be found in his thesis (Ebner 1998). The transformations consisted of two 40x40x40 node LUTs that spanned an input CIELAB range of a* and b* of [-128 127]. The main difference between these tables and those generated from the Hung and Berns data set (Section
3.2.1) were their lightness dependence and the larger number of data points in the Ebner and Fairchild data set.

### 3.3 Experimental Testing

Testing the hue-linearity properties of CIELAB and the two hue-corrected spaces consisted of a paired-comparison psychophysical experiment. Observers viewed pairs of pictorial images that were gamut mapped in CIELAB and the hue-corrected color spaces. In this experiment the images were viewed on a colorimetrically controlled monitor with a white point set to chromaticities near CIE Illuminant D65. Device-dependent image pixel data (digital counts) were converted to and from CIELAB using a gain-offset-gamma CRT characterization model (GOG) (Berns, Motta, and Gorzynski (1993)). The results of this experiment are compared to those obtained by Ebner (1998) in his thesis for a similar experiment that considered hue leaves rather than images.

#### 3.3.1 Image Processing Path

For this experiment, the viewing conditions were set to chromaticities of D65. This was done so that the viewing conditions would be optimized for CIELAB. The CIELAB color space was defined for a reference illuminant of D65. For viewing conditions specified for a different source, it would be necessary to convert the reference tristimulus values to corresponding D65 matches. This could be accomplished by using a color-appearance model such as CIECAM97s.
The CIELAB hue-correction LUTs were applied to the image pixel data. Then all gamut-mapping operations were applied. The image data were then processed through the inverse LUTs, back into CIELAB.

### 3.3.2 Hue-Leaf Experiment

The hue-leaf experiment performed by Ebner (1998) consisted of converting 15 hue planes, uniformly sampled in lightness and chroma, to CIELAB from the two hue-corrected color spaces. In addition, CIELAB constant metric hue planes were generated for the same 15 hue angles. CIELAB points outside the CRT gamut were converted to a neutral gray (CIELAB coordinates L*a*b* = [50,0,0]). Only same-base-hue images were compared to each other. In a paired-comparison experiment the observers were asked to pick the hue plane that was the most uniform.

### 3.3.3 Pictorial Gamut-Mapping Experiment

The gamut mapping experiment consisted of gamut mapping 5 full-gamut CRT images to an inkjet printer gamut scaled to fit within the full CRT gamut so that the visual experiments could be performed displaying the originals and the reproductions on the monitor. This made it much easier to prepare the images and perform the experiment. Three gamut-mapping algorithms were used: 1.) chroma clipping with lightness and hue preservation, 2.) minimum ΔE*ab clipping with hue preservation, and 3.) centroid clipping (i.e., radial clipping toward L*a*b* = [50,0,0] with hue preservation). These algorithms
were selected to be realistic and representative gamut mapping transformations, and for their ease of computation. This experiment was designed to test the hue uniformity characteristics of the color spaces, not the "goodness" of the particular gamut mapping algorithms. Five images were selected to span many different hue regions. The images used in this experiment are shown in Appendix F.

The gamut-mapped images were shown to 22 observers in sets of three images. The three images were a full-gamut original and two reduced-gamut reproductions. Only reproductions that were created using the same gamut-mapping algorithm were compared. The idea was to test the color spaces, not to evaluate the differences among the gamut-mapping techniques. A paired-comparison technique was used, and the gamut-mapped reproductions consisted of all possible pairs of the three color spaces for each scene. The interface allowed observers to view only one image at a time. They were allowed to freely toggle among the three images. They were asked to pick the reproduction that was closest in hue to the original full gamut image.

The visual data from the hue leaf and pictorial image experiments were used to generate interval scales using Thurstone's law of comparative judgment (Thurstone (1927), Bartleson and Grum (1984)).
3.4 Experimental Results

3.4.1 Hue-Leaf Experiment Results

The results of the hue-leaf experiments are given by the interval scale results in Figure 3-5. At a given base CIELAB hue angle, a significant difference between two color spaces exists when the 95-percent error bars of one space do not bound the interval scale value of another. For example, at the $h_{ab}=0^\circ$ leaf it was not possible to say that one of the color spaces produced a leaf that was more uniform than either of the other spaces did. However, at the $h_{ab}=288^\circ$ leaf, it was possible to say that both of the hue-corrected color spaces produce significantly more uniform leaves than CIELAB. It was not possible, however, to say which of the hue-corrected spaces was more uniform. Overall, when the two hue-corrected spaces were deemed significantly more uniform than CIELAB they were judged to be equally uniform.

Surprisingly, the results of the hue-leaf experiment indicated that CIELAB was as uniform as, or more uniform than, either hue-corrected space. Exceptions for this were found in the “blue” region of the color space (i.e., approximately $h_{ab} = 260^\circ$ to $300^\circ$). In this region the hue leaves generated using the hue-corrected color spaces were judged to be significantly more uniform than the hue leaves generated in the CIELAB color space.
Figure 3-5. Interval scales for the 15 hue leaves. In order for one space to be significantly better than another, the mean of one space has to be outside the error bars of the other.

3.4.2 Pictorial Gamut-Mapping Experiment Results

The results of the pictorial gamut mapping experiment showed that, over all of the images and routines tested, the hue-linearized versions of CIELAB maintained the perceived hue of the gamut-mapped images better than CIELAB, Figure 3-6. In addition, these scales tend to indicate that, overall, the Hung and Berns hue-corrected CIELAB space slightly out-performs the Ebner and Fairchild hue-corrected CIELAB space. (Note: The interval scales for the individual images are given in Appendix G.)
Figure 3-6. Interval scale pooled over all images and gamut mapping routines tested. Results show that Hung and Berns hue-corrected CIELAB outperforms the Ebner and Fairchild hue-corrected CIELAB and CIELAB.

Because the hue-leaf experiment found that the hue-linearization performed by the Hung and Berns and Ebner and Fairchild LUTs was not better than CIELAB for all hue angles, the analysis of the pictorial gamut mapping experiment was divided into dominant image color. While the images contained colors that spanned CIELAB, there were specific dominant features in the images that were most sensitive to hue shifts. The images were broken down into three categories: dominantly red images, dominantly blue images, and mixed images. In doing this, the results of the pictorial gamut mapping experiment were very similar to those found in the hue-leaf experiment.
When the observer data were analyzed using these categories, the following conclusions were made. For images that were classified as having predominantly red features (i.e., the “mushroom” and “macaws” images), the data supports using CIELAB as the color space for gamut mapping, Figure 3-7. Images that contained predominantly blue features (i.e., the “capital” and “pool balls” images) were gamut-mapped better using the hue-corrected CIELAB space generated by the Hung and Berns data set, Figure 3-8. The Ebner and Fairchild LUT also performed well for these images. Finally, the image that contained a mixture of red, green, blue, and yellow features (i.e., the “Amsterdam” image), the results indicate that the hue-corrected spaces were selected as the most hue-preserving, Figure 3-9. The reason is that it had significant number of “blue” features that were shifted toward purple as a result of the chroma compression when the CIELAB color space was used.
Figure 3-7. Interval scale results for the images that contained predominantly red features that were gamut mapped. Results show that performing the gamut mapping in the CIELAB space maintained the hue of the original image better than either hue-corrected space.

Figure 3-8. Interval scale results for the images that contained predominantly blue features. Results indicate that the hue-corrected versions of CIELAB more accurately preserved the original hue of the image after gamut mapping.
3.5 Conclusions

Constant-visual-hue data were used to generate LUTs to linearize the CIELAB space with respect to hue. These hue-corrected CIELAB spaces were then used in a gamut mapping experiment to evaluate whether the hue-corrected spaces preserve the hue of the original scene better than CIELAB. The results of these experiments showed the benefit of using the hue-corrected CIELAB space for blue features. The reason that the CIELAB color space performed so poorly for blue colors was that it caused a hue-name change for these colors as they were reduced in chroma. In the hue-corrected spaces, the color name remained constant throughout the entire chromatic range.
The hue of image features that were outside the blue region of color space (\(h_{ab} = 260^\circ\) to 300\(^\circ\)) were not preserved better using the hue-correction LUTs tested in these experiments. These results led to the generation of a new set of LUTs that provided hue-correction only in the "blue" region of CIELAB, where the strongest hue-nonlinearity exists. Illustrations of these transformations are given in Figure 3-10 and Figure 3-11. These hue-correction LUTs were used in the remaining gamut mapping experiments performed in this thesis. (Note: These LUTs are given in Appendix H.)

![Graph](image)

*Figure 3-10. Forward hue-correction LUT with only correction in the "blue" region generated from the Berns and Hung (1995) data set.*
3.6 Recommendations

Further research in this area should consider testing the IPT color space (Ebner and Fairchild (1998b)) for a series of gamut mapping algorithms. It would be a significant advancement to gamut mapping if this space was shown to have good performance characteristics for image-reproduction tasks. In addition, it would be interesting to use some of the visual data from Ebner and Fairchild and Hung and Berns to straighten hue lines in the CIECAM97s color space. This correction could be in the form of multidimensional LUTs or in a modification to the color appearance transforms themselves.
4 Image Lightness Rescaling Using Sigmoidal Contrast Enhancement Functions

4.1 Summary

In color gamut mapping of pictorial images, the lightness rendition of the mapped images plays a major role in the quality of the final image. For color gamut mapping tasks, where the goal is to produce a match to the original scene, it is important to maintain the perceived lightness contrast of the original image. Typical lightness remapping functions such as linear compression, soft compression, and hard clipping reduce the lightness contrast of the input image. Sigmoidal remapping functions were utilized to overcome the natural loss in perceived lightness contrast that results when an image from a full dynamic range device is scaled into the limited dynamic range of a destination device. These functions were tuned to the particular lightness characteristics of the images used and the selected dynamic ranges. The sigmoidal remapping functions were selected based on an empirical contrast-enhancement model that was developed from the results of a psychophysical-adjustment experiment. The results of this study showed that it was possible to better maintain the perceived lightness contrast of the images by using sigmoidal contrast enhancement functions to selectively rescale images from a source device with a full dynamic range into a destination device with a limited dynamic range.
4.2 SIGMOIDAL LIGHTNESS RESCALING FUNCTIONS

The biggest limitation with the lightness remapping strategies previously discussed is that they fail to universally address the fact that as the dynamic range decreases the perceived contrast of the image decreases. As such, it is desirable to develop a remapping strategy that will perform the range compression while maintaining the perceived image contrast. The proposed solution to this problem was to develop a function that would be tunable such that as the dynamic range decreased, the function would boost the image contrast accordingly. In order to boost the image contrast in the limited dynamic range, both the highlight and the shadow detail need to be compressed. This was accomplished by utilizing a sigmoidal remapping function. Recently, Holm (1996a,b) proposed a scene dependent mapping process for a digital camera system that utilized sigmoidal functions. His transformations were based on the well-known photographic Zone system proposed by Ansel Adams (Stroebel et al. (1986)). In Holm's process he related specific input scene log luminances, Zones I (black) and IX (white), to specific output density Zones possible for different reproduction media. He then utilizes the Zone characteristics of the input scene and the output media to develop a sigmoidal mapping function whose "flex" was determined from the dynamic range differences between the devices and a "shift" that was based on the mean-log luminance of the scene. This process results in image/media-dependent remapping functions. His conclusions were that these functions produced "preferred" reproductions similar to those achieved through well-known
photographic processes that trade accurate tone reproduction for a preferred tone reproduction. This rendering goal differs from that established for this study (i.e., accurate lightness appearance matches).

The form of the sigmoidal functions was derived from a discrete cumulative normal function (S), given in Equation 4-1, where \( x_o \) and \( \Sigma \) are the mean and standard deviation of the normal distribution respectively, \( i = 0,1,2...m \), and \( m \) is the number of points used in the discrete look-up table (LUT). In order to use \( S \) as a lightness remapping LUT \( (S_{LUT}) \) it must first be normalized into the lightness range of [0 100]. These normalized data are then scaled into the dynamic range of the destination device, as given in Equation 4-2, where \( L^*_{minOut} \) and \( L^*_{maxOut} \) are the black-point and white-point lightnesses of the destination device respectively.

\[
S_i = \sum_{n=0}^{n=i} \frac{1}{\sqrt{2\pi \Sigma}} e^{-\frac{(100n-m-x_o)^2}{2\Sigma^2}} 
\]

\[
S_{LUT} = \frac{(S_i - \min(S))}{(\max(S) - \min(S))} (L^*_{maxOut} - L^*_{minOut}) + L^*_{minOut}
\]

The \( x_o \) and \( \Sigma \) parameters control the shape of the sigmoid. The value of \( x_o \) controls the centering of the sigmoid and \( \Sigma \) controls the slope. Making \( x_o \) greater than \( L^*=50 \) shifts the straight-line portion of the sigmoid toward higher lightnesses. An \( x_o \) value of less than \( L^*=50 \) shifts the straight-line portion of the sigmoid toward the lower lightnesses. These relationships are shown in Figure 4-1a where \( \Sigma=15 \). In a similar manner, a family of
sigmoidal remapping curves can be generated by holding the \( x_0 \) parameter fixed, at \( L^* = 50 \) for example, and varying the \( \Sigma \) parameter. Figure 4-1b. Decreasing the \( \Sigma \) value has the effect of increasing the contrast of the remapped image. Shifting the distribution toward lower lightnesses (i.e., decreasing \( x_0 \)) has the effect of applying more highlight compression, while shifting the distribution toward higher lightnesses (i.e., increasing \( x_0 \)) results in more shadow compression. By adjusting \( x_0 \) and \( \Sigma \) it is possible to tailor a remapping function with an appropriate amount of image-contrast enhancement and highlight and shadow lightness compression.

![Figure 4-1](image)

**Figure 4-1.** a.) Family of sigmoidal contrast enhancement functions that have equal \( \Sigma \) parameters and varying \( x_0 \) parameters. As the \( x_0 \) parameter increases there is more compression of the shadow detail than the highlight detail. b.) Family of sigmoidal contrast enhancement functions that have equal \( x_0 \) parameters and varying \( \Sigma \) parameters. As the \( \Sigma \) parameter is decreased the remapping function increases the image contrast by boosting the slope in the mid-tones while equally compressing the highlight and the shadow detail.
4.3 SIMULTANEOUS LIGHTNESS CONTRAST

The hypothesis of using sigmoidal functions for lightness remapping is based on the phenomenon of simultaneous lightness contrast. It is possible to make the dark colors in an image look darker by making the light colors lighter. This is accomplished using the sigmoidal functions. As $\Sigma$ decreases the effect is to lighten the highlights and darken the shadowed regions. As the lightness difference between the highlight and shadow regions increases, the image contrast increases giving the appearance of a larger dynamic range. By adjusting $x_0$ and $\Sigma$ together, it is possible to tailor the amount of lightening and darkening of the highlight and shadowed regions to control the overall contrast enhancement.

4.4 EXPERIMENTAL

4.4.1 Phase 1 - Visual optimization of $x_0$ and $\Sigma$: User adjustments

Typical pictorial images contain a range of shadow and highlight detail depending on the composition of the scene portrayed in the image. Information regarding the lightness composition of the image can be obtained from its lightness histogram. For the purposes of this study, lightness histograms were broken down into four categories: low-lightness key (skewed toward low lightness), high-lightness key (skewed toward high lightness), normal-lightness key ("Gaussian" shaped histogram), and uniform lightness key ("flat" histogram over most of the lightness range). (These cases are similar to the "four basic
image types” given by Gonzalez and Woods (1992) pp. 174.) It was theorized that the form of the sigmoidal lightness-remapping function would depend on the lightness composition of the image (i.e., described by its lightness histogram) as well as the dynamic range difference between the source and the destination devices. MacAdam (1951) showed that different sigmoidal density tone transfer curves were required for optimal tone reproduction of images that had predominantly shadow or highlight features.

Based on these observations, a psychophysical experiment was conducted to determine optimal sigmoidal contrast enhancement functions for images from each of the lightness keys mentioned above, at four destination dynamic ranges (\(L^*_{\text{minOut}} = \{5, 10, 15, 20\}\)). This experiment consisted of user adjustments that were performed on six monitor images that were selected based on their lightness histograms (Figure 4-3a-f). These included one high-lightness key image, one low-lightness key image, two normal-lightness key images, and two uniform-lightness key images (Figure 4-2a-f). (Lightness images were generated using the gain-offset-gamma characteristics of the Sony GDM-2000TC monitor to convert from device digital counts to CIELAB \(L^*\), according to the model established by Berns, Motta, and Gorzynski (1993).) The adjustments consisted of having users interactively control the shapes of the sigmoidal lightness (CIELAB \(L^*\)) remapping functions. In this experiment, only the calibrated grayscale lightness channel was presented to the observers for each image. This made it possible for the observers to interactively remap the lightness image in real time. The premise behind adjusting only
the lightness image was that the lightness information could be visually separated from the chromatic information. In addition, it was hypothesized that the optimal lightness remappings that were obtained from the grayscale adjustments would be the same optimal lightness remappings when the chromatic content of the scene was added back in. (This was confirmed in Phase 2 and Phase 3 where the chromatic content was added back.) For each observer, the adjustments resulted in optimal settings for the sigmoidal rescaling curves.

Figure 4-2a-f. Lightness images for the six test images used in the user adjustment experiments; a.) "Couple-On-Beach"; b.) "Horse-Race"; c.) "Flowers"; d.) "Macaws"; e.) "Temple"; f.) "Raft". (Images a-f proceed from left to right and top to bottom.)
The user interface developed for this experiment is shown in Figure 4-4. Subjects controlled sliders that dynamically updated the values for $x_0$ and $\Sigma$. In turn, the form of
the sigmoidal-remapping function was adjusted accordingly. The task presented to the subjects was to adjust the image in the right field (the reduced dynamic range condition) until it best matched the image in the left field (the original full dynamic range condition). For this experiment, adjustments were made for each of the six images at four different L* minOut levels (L* minOut = 5, 10, 15, and 20). In all, six subjects took part in the adjustment experiment. The subjects were experienced with these types of adjustments.

Figure 4-4. User interface for adjustment experiment. The subjects adjusted the x₀ and Σ sliders until the reduced dynamic range image (right image) was the best possible match to the original full dynamic range image (left image).
The results of the observer adjustments are represented graphically in Figure 4-5a-f. Based on the large inter-observer variability obtained from these adjustments, an "average-observer" response was generated from the numerical average of the individual adjustments. The solid trend lines shown in Figure 4-5a-f represent the average-observer responses. Overall, as the minimum lightness of the destination device increased, the average-observer $x_0$ parameter increased and the average-observer $\Sigma$ decreased. This trend indicates that as the dynamic range decreased more compression was required in the low lightness region, as evidenced by the increase $x_0$. Similarly, the trend lines for the $\Sigma$ parameter indicate that a contrast boost (decrease in $\Sigma$) is required as the dynamic range decreases.

4.4.2 Phase 2 - Selection of candidate remapping curves

The user adjustment results shown in Figure 4-5a-f indicate a significant amount of inter-observer variability. One probable reason for the large variability in the observer adjustments came from preferential weighting of different regions in the image. Since the dynamic ranges of the input and output displays were different and the images were displayed side-by-side, it was not always possible to generate exact matches between the original, high dynamic range device, and the reduced range reproduction. As such, the observers had to decide which regions of the scene were the most important to preserve. In doing so, the form of the reproduction curve that was the best for one region may not have produced the best match for another region.
Based on the large inter-observer uncertainty, it was decided that the "average observer" curves should be used only as candidates for the optimal $x_0$ and $\Sigma$ parameters. Therefore, for each of the six test images, a family of curves was generated for each $L^*_{\text{minOut}}$ level. The $x_0$ and $\Sigma$ settings for these curves were determined based on the standard deviation of the inter-observer variability at each of the $L^*_{\text{minOut}}$ adjustment levels. For example, for the "Macaws" image, at a minimum $L^*=10$, there were six estimates of $x_0$ (i.e., one from each observer). The mean of these $x_0$ settings made up one point in the "average observer" trend line for that image. Two other estimates were then made of $x_0$ by taking $x_0 \pm \sigma_{x_0}$, where $\sigma_{x_0}$ was the standard deviation of the inter-observer variability for $x_0$ (as shown by the error bars on Figure 4-5a-f). The same process was used to generate three estimates of $\Sigma$ at each minimum $L^*$ level for each image. Given the three estimates for $x_0$ and the three estimates for $\Sigma$ there were nine candidate contrast enhancement curves ($S_1$, $S_2$, ... $S_9$), at each minimum $L^*$ level, for each of the six reference images (i.e., $S_1=f(x_0, \Sigma), S_2=f(x_0, \Sigma+\sigma_{\Sigma}), S_3=f(x_0, \Sigma-\sigma_{\Sigma}), S_4=f(x_0+\sigma_{x_0}, \Sigma), S_5=f(x_0+\sigma_{x_0}, \Sigma+\sigma_{\Sigma}), S_6=f(x_0+\sigma_{x_0}, \Sigma-\sigma_{\Sigma}), S_7=f(x_0-\sigma_{x_0}, \Sigma), S_8=f(x_0-\sigma_{x_0}, \Sigma+\sigma_{\Sigma}), S_9=f(x_0-\sigma_{x_0}, \Sigma-\sigma_{\Sigma})$).

In order to determine which of these nine candidate-remapping functions produced the best match to an original image for a given destination $L^*_{\text{minOut}}$, a psychophysical test was performed. Thus, for each image, 36 lightness-compressed images were generated (i.e., 4 minimum $L^*$ levels times 9 candidate sigmoids per level). Since the ultimate goal of this research was to apply the sigmoidal lightness compression on color images, the lightness
compressed images were recombined with their corresponding hue and chroma data (i.e., CIELAB \( h_{ab} \) and \( C_{ab}^* \) respectively). Thus, the remapped images were identical in hue and essentially identical in chroma to the original. (Note: The lightness remapping of a pixel may have moved that pixel’s color out-of-gamut. As a result, after the lightness remapping, all pixels that were out-of-gamut were chroma clipped to the surface of the gamut while preserving lightness and hue angle. These mappings were performed in the Hung and Berns hue-linearized CIELAB color space (Braun, Ebner, and Fairchild (1998)).)
Figure 4-5a-f. Sigmoidal parameter curves from the adjustment experiment. These plots give the $x_0$ and $\Sigma$ settings that each of the six subjects determined produced a visual match, to the original full dynamic range image, under the four reduced dynamic range conditions. The solid lines in each plot represent the trend lines that connect the average $x_0$ and $\Sigma$ parameters taken from the individual subject responses.
The following procedure was then performed on the full-color images to eliminate the obviously poor performing remapping functions. One observer (the author) performed this task since for six images, with four minimum L* levels per image, and nine sigmoid settings per level, the number of pairs exceeded 800. This was far too many image-pair comparisons for multiple subjects. The subject that performed this task was experienced in these types of observations. Since there was only one observation per image, the paired comparison data could not be analyzed using Thurstone's law of Comparative Judgements (Bartleson and Grum (1984)). As such, the results of these observations were analyzed by tallying the number of times a given setting was selected as the best. Thus, at each $L^{*}_{\text{minOut}}$ setting, for each image, there was a histogram of number of times a given setting ($S_1-S_9$) was selected as a better match. For example, for the "Temple" image at $L^{*}_{\text{minOut}}=10$, there were nine images mapped through their corresponding $S_1-S_9$ sigmoidal-remapping functions. These images were compared in pairs to the original. Each remapped image was compared to the eight other remapped images at that $L^{*}_{\text{minOut}}$ value. A tally was taken of the number of times a given image was selected as the better match to the original. The tallied data from these observations are given in Table I-1 (Appendix I).
4.4.3 Phase 3 - Selection of optimal rescaling curves

The highlighted images in Table I-1 (Appendix I) advanced into the third visual experiment. These final candidate images were shown, in pairs, to 21 observers. Their task was to select the image that was the closest match to the original scene. The results of this experiment were 24 interval scales (Bartleson and Grum (1984)) (i.e., one for each of the four minimum L* levels, for each of the six images) that were used to select an optimal pair of $x_o$ and $\Sigma$ values. The interval scales are shown in Table J-1 (Appendix J). The $x_o$ and $\Sigma$ values associated with the highlighted settings were selected as the optimal-sigmoid parameters for the different images and $L^*_{\text{minOut}}$ levels. In general, the optimized parameters followed the same trends as with the average-observer curves: as the $L^*_{\text{minOut}}$ increased the amount of contrast boosting increased.

For some of the images there were dips present in the $x_o$ and $\Sigma$ parameter curves. For example, in the "Couple-on-Beach" image the $x_o$ and $\Sigma$ values for the minimum $L^*=15$ level were significantly lower than for any of the other settings, Figure 4-6. The $x_o$ and $\Sigma$ values at minimum $L^*=15$ were not in line with the values at the other $L^*_{\text{minOut}}$ settings. As such, the $x_o$ and $\Sigma$ parameters for this level were increased until they fell naturally in line with the other settings for this image. Intuitively there should not be discontinuities in the remapping curves between minimum $L^*$ levels of 10 and 15, or of 15 and 20 within an image. The reason for this is that the scene content did not change; only the output dynamic range changed.
One possible explanation for these results is that the range of values covered by the mean plus/minus the standard deviation of the inter-observer variability did not contain both the $x_o$ or the $\Sigma$ values that were in line with those presented at the other minimum lightness values. As such, the observers were not presented an image that was mapped using the same trends as the other black-point levels. This was due to choice of the settings that were used in the experiment. Similar adjustments were made to the parameter curves for the "Macaws", "Raft", and "Temple" images. Upon visual inspection, these adjustments produced images that were equal, if not superior, in quality to the unadjusted curves. The final forms of the "optimal" sigmoid parameter curves are given in Figure 4-7a-c. These curves have been grouped together based on their similarity.

![Figure 4-6. The solid lines represent the $x_o$ and $\Sigma$ parameters resulting from the Phase 3 visual experiment. The "dip" in the estimated parameters for the $L^*_{\text{minOut}}=15$ setting was adjusted so that it fell in line with the settings at the other $L^*_{\text{minOut}}$ levels (dashed lines).]
Figure 4-7a.

Figure 4-7b.
4.5 ANALYSIS OF PHASE 3 $x_0$ AND $\Sigma$ PARAMETER CURVES

4.5.1 Image Groupings

Analysis of the form of the $x_0$ and $\Sigma$ parameter curves for the six images revealed that there was considerable correlation between several images (Figure 4-7). Originally the images were classified into four groups; low-lightness key, high-lightness key, normal-lightness key, and uniform-lightness key. The results of this experiment tend to indicate that, of the images tested, three lightness classes were enough to categorize the images; these classes were high-lightness class, normal-lightness class, and low-lightness class.
As such, the "Couple-On-Beach" image and the "Temple" were grouped together into the high-lightness class because their corresponding $x_o$ and $\Sigma$ parameter curves were nearly identical. The new normal-lightness class consisted of the "Flowers", "Macaws", and "Raft" images. The low-lightness class consisted only of the "Horse-Race" image since it was considerably different than any of the other images.

4.5.2 Analysis of Lightness-class Groupings

For the high-lightness class, the "Couple-On-Beach" and "Temple" images resulted in nearly identical $x_o$ and $\Sigma$ parameter curves because they both had a significant amount of highlight information. This was revealed by examining the cumulative-lightness histograms for these images (Figure 4-3a,e). For these images, the 75-percent points of their cumulative-lightness histograms occurred at lightness values of greater than 70. The sigmoidal-remapping functions used in this study compress both the highlight and shadowed regions to increase the perceived-image contrast (i.e., simultaneous-lightness contrast). Essentially, the shadowed regions are made to appear darker by compressing them while simultaneously lightening the highlight regions. Since these images contained proportionately the same amount of highlight regions, the same $x_o$ and $\Sigma$ parameter curves performed well for them.

Similar trends in the $x_o$ and $\Sigma$ parameter curves were noticed for the three images that were grouped into the normal-lightness class ("Flowers", "Macaws", and "Raft"), Figure
4-7b. The 75-percent points of their cumulative-lightness histograms occurred at 55, 51, and 50 lightness units for the "Flowers", "Macaws", and "Raft" images respectively (Figure 4-3c,d,f). These three images had nearly identical Σ curves and had x₀ curves that looked liked shifted copies of each other. The similarity in the proportionate amounts of highlight and shadow detail point to the similarity in the x₀ and Σ curves for these images.

Finally, in the low-lightness class category, the "Horse-Race" image had the least amount of highlight detail and the most amount of shadow detail. This was indicated by its the rapidly rising cumulative-lightness histogram shown in Figure 4-3b. The corresponding 75-percent point of its cumulative-lightness histogram occurred at L* = 31. Thus, 75-percent of the entire image pixels occur in essentially one third of the entire lightness-dynamic range. In this case the x₀ and Σ parameter curves apply more highlight compression than shadow compression.

4.5.3 Curve Consolidations

The parameter curves were grouped together into three distinct lightness classes in an effort to consolidate the individual image x₀ and Σ parameter curves into a single parameter curve that described these parameters for the entire class. Based on the high correlation between the x₀ and Σ parameter curves in the high-lightness class (Figure 4-7a) it was possible to use the x₀ and Σ parameter curves for the "Couple-on-Beach" image to predict the contrast enhancement needed for the "Temple" image. When this
was done, there were no significant changes noticed in the appearance of the mapped "Temple" image. The plots shown in Figure 4-8a,b give the final form of the high-lightness class $x_o$ and $\Sigma$ parameter curves.

Similarly, correlation was found between the normal-lightness key images, "Macaws" and "Flowers", and the uniform-lightness key "Raft" image, Figure 4-7b. The $\Sigma$ curves for these three images were nearly identical. The exception was for high minimum $L^*$ settings. In this case the $\Sigma$ curve for the "Macaws" image had an essentially linear, decreasing, relationship as a function of minimum $L^*$. The $\Sigma$ curves for the other two images were essentially linear from minimum $L^*=5$ to 15 and then leveled out for minimum $L^*=20$. The reason for this was that there was slightly more information in the "Flowers" and "Raft" images at lower lightness values. Since the amount of low-end compression increased with $L^*_{\text{minOut}}$, the contrast boost associated with decreasing $\Sigma$ and increasing $x_o$ needed to level off for the "Raft" and "Flowers" images. This helped to insure that the low-end shadow detail did not get compressed to the point where all of the lightness contrast was eliminated. Based on the high correlation that existed for the $\Sigma$ parameter curves, a composite $\Sigma$ curve was generated for the normal-lightness class. This curve followed a nearly linear decrease in $\Sigma$ up to $L^*_{\text{minOut}}=15$ and then flattened out to a value at $L^*_{\text{minOut}}=20$ of the average of the parameters from the three normal-lightness key images, Figure 4-8b.
The systematic differences between the $x_0$ curves for the three images (Figure 4-7b) in the normal-lightness class made it difficult to consolidate them into a single $x_0$ parameter curve for the class. A ranking of these curves (e.g., highest, middle, and lowest) correlated with the lightness of the 75-percent point of their respective cumulative-lightness histograms. The rankings of the $x_0$ curves was highest="Flowers", middle="Macaws", lowest="Raft". The lightness values for the 75-percent point of the images cumulative-lightness histograms were 55, 51, and 50 for the "Flowers", "Macaws", and "Raft" images respectively. The $x_0$ parameter curves for the normal-lightness key images "blended" very smoothly between the shape of the $x_0$ curves for the low-lightness class and the high-lightness class. The "Raft" image had an $x_0$ curve that was similar to that of the low-lightness class. The 75-percent point of the "Raft" image was the closest to the "Horse-Race" image for all of the images in normal-lightness class. The $x_0$ curve for the "Flowers" image was similar in shape and magnitude to the high-lightness class. The value of the 75-percent point for this image was the highest of the images in the normal-lightness class. The "Macaws" image fell in between the "Raft" and the "Flowers" image in both $x_0$ curve shape and 75-percent point of the cumulative-lightness histograms. Based on these trends in the $x_0$ curves, the "Macaws" $x_0$ curve was selected to represent an average normal-lightness class image, Figure 4-8a.

To summarize, for each of the image lightness classes $x_0$ and $\Sigma$ parameter curves were selected to represent average images from that class. For the high-lightness class, the $x_0$
and $\Sigma$ parameter curves were selected from the "Couple-on-Beach" image. The $x_\alpha$ curve that represented the normal-lightness class was taken from the "Macaws" image. The $\Sigma$ parameter curve for the normal-lightness class was made from a composite of the three images in the class. The $x_\alpha$ and $\Sigma$ parameter curves for the low-lightness class were taken from the "Horse-Race" image. These curves are shown in Figure 4-8a-b. The resulting sigmoidal-remapping functions derived from these parameters are shown in Figure 4-9a-c. The remapping functions shown in Figure 4-9a-c have been normalized between 0 and 100 to illustrate the differences in contrast as well as differences in high and low lightness level compression. In practice these remapping functions would have an input range from $L^{*}_{\text{in}} = [0 \; 100]$ and an output range from $L^{*}_{\text{out}} = [L^{*}_{\text{min}} \; 100]$. The form of the remapping functions given in Figure 4-9a-c indicate that in order to maintain the contrast of the original scene, in a reduced dynamic range condition, the contrast of the image must be increased. In addition, the fact that the different image classes resulted in distinctly different shaped remapping functions indicates that the selection of an optimal lightness-scaling function is image dependent.
Figure 4.8a,b. Final \( x_0 \) and \( \Sigma \) parameter curves for the high, normal, and low-lightness classes.
Figure 4-9a.
Figure 4-9b.
Figure 4-9a-c. Sigmoidal contrast enhancement remapping functions for high (a), normal (b), and low (c) lightness-classes. The curves have been normalized over the range of $L^* = [0 \ 100]$ to illustrate the difference in contrast of the remapping functions as the dynamic range decreases. In practice, the input lightness range would be between [0 100] and the output lightness range would be between $[L^*_{minOut} \ 100]$. 
4.6 **EMPIRICAL SIGMOID MODEL**

Based on the results of Phase 3, it was possible to construct an empirically based model that allows for the automatic selection of \( x_0 \) and \( \Sigma \) parameters. The selection process is based on first identifying the lightness of the 75-percent point of the cumulative-lightness histogram and the black point of the destination gamut. These parameters are then used to derive the sigmoidal parameters \( (x_0, \Sigma) \). The \( x_0 \) and \( \Sigma \) parameters used in this model are given in Table K-1 (Appendix K). The sigmoidal parameters are selected using a sequential linear-interpolation process from the optimal curves generated in Phase 3 (Figure 4-8a,b).

The sequential-interpolation process for \( x_0 \) is illustrated in the following steps using the example where the input image has a 75-percent point of its cumulative-lightness histogram at \( L^* = 55 \) and an \( L^*_{\text{minOut}} = 18 \). The steps involved in the sequential-linear interpolation for \( x_0 \) are shown in Figure 4-10 and are given by the following: (Note: \( \Sigma \) is calculated in the same manner from the \( \Sigma \) parameter curves shown in Figure 4-8b).

**Interpolation Steps for \( x_0 \):**

1. Specify the minimum lightness \( (L^*_{\text{minOut}}) \) of the destination device.

2. Determine the lightness of the 75-percent point of the cumulative-lightness histogram for the test image.

122
3. Determine which parameter curves to use for the interpolation. Compare the lightness of the 75-percent point of the test image to that of the 75-percent points of the reference lightness (high-lightness class = 71 L* units, normal-lightness class = 51 L* units, and low-lightness class = 31 L* units). If the L* associated with the 75-percent point of the input cumulative histogram is greater than 71 or less than 31, the high or low-lightness class parameter curves are used respectively.

For example, if the test image has a 75-percent point lightness of L*=55 and an L*_minOut=18, then this image is bounded on the upper end by the high-lightness class image curve (L*=71) and the lower end by the normal-lightness class image curve (L*=51). As such, subsequent interpolations are performed using these curves as references.

4. Estimate x₀ parameters for the test L*_minOut level by linearly interpolating between the reference L*_minOut levels of {5,10,15,20} for the current lightness-class curves.

For the current example, the test L*_minOut equals 18. The x₀ parameters for the upper and lower bounding lightness-class curves (i.e., x₀High(18) and x₀Normal(18)) are estimated at an L*_minOut=18 from the x₀ values at the
corresponding $L_{minOut}^*$ values of 15 and 20. These relationships are given by:

$$\begin{align*}
x_{oHigh}(18) &= \alpha x_{oHigh}(15) + \beta x_{oHigh}(20) \\
x_{oNormal}(18) &= \alpha x_{oNormal}(15) + \beta x_{oNormal}(20)
\end{align*}$$

(4-3)
(4-4)

where $x_{oHigh}(18)$ and $x_{oNormal}(18)$ are the estimated $x_o$ values for an $L_{minOut}^* = 18$ for the high- and normal-lightness classes respectively and $x_{oHigh}(15)$, $x_{oHigh}(20)$, $x_{oNormal}(15)$, and $x_{oNormal}(20)$ are the $x_o$ model values for the high- and normal-lightness classes given in Table K-1 (Appendix K). The interpolation weights given by $\alpha$ and $\beta$ are determined by:

$$\begin{align*}
\alpha &= 1 - \beta \\
\beta &= \frac{18 - 15}{20 - 15}
\end{align*}$$

(4-5)
(4-6)

where the values of 18, 15, and 20 are the corresponding $L_{minOut}^*$ values for the $x_o$ parameters used in the interpolation shown in Equations 4-3 and 4-4.

5. Estimate final $x_o$ parameter ($x_{oEstimated}$) for the current image by linearly interpolating between the $x_o$ points estimated in step 4 using the cumulative-histogram points for the bounding lightness-classes and that of the current image as weights. The weighting equation used in this calculation is given by:
\[ x_{\text{Estimated}} = \omega_1 \cdot x_{\text{High}(18)} + \omega_2 \cdot x_{\text{Normal}(18)} \]  

(4-7)

where \( x_{\text{High}(18)} \) and \( x_{\text{Normal}(18)} \) come from Equations 4-3 and 4-4. The interpolation parameters \( \omega_1 \) and \( \omega_2 \) for this example are given by:

\[ \omega_1 = 1 - \omega_2 \]  

(4-8)

\[ \omega_2 = \frac{55 - 51}{71 - 51} \]  

(4-9)

where the values of 55, 51, and 71 are the corresponding lightness values for the 75-percent points of the cumulative-lightness histograms for the test image, the normal-lightness class, and the high-lightness class respectively.
Figure 4-10. Example interpolation of a $x_0$ parameter for an input image with its 75-percent point cumulative-lightness histogram point at $L^*=55$ and an $L^\text{outMin}_\text{in} = 18$. The "$x_0$ Estimated" point is calculated using linear interpolation between the model parameter points located at $L^\text{outMin}_\text{in}$ values of 15 and 20, using the weighting equations given in Equations 4-3 and 4-4. The $x_0$ and $\Sigma$ parameters used in these interpolations are given in Table K-1 (Appendix K).

4.7 CONCLUSIONS

Based on the results of the experiments performed in this study, it was possible to maintain a large portion, if not all, of the perceived contrast of lightness-compressed
images by increasing the image contrast using sigmoidal contrast-enhancement curves
before or during the compression process. In general, the form of the optimal-sigmoidal
functions was image dependent, directly linked to the lightness histogram of the input
images, and the black point lightness of the destination devices. The form of the
enhancement curves for an arbitrary input image was determined based on a simple series
of interpolations from a set of optimized-reference curves. The only inputs to this process
were the lightnesses of the source and destination black points and the lightness
corresponding to the 75-percent point of the cumulative-lightness histogram of the image.
It is believed that these functions will perform a crucial role in developing a more
universal approach to color gamut mapping of pictorial images.
5 Testing of Gamut-Mapping Algorithms

5.1 Summary

Gamut-mapping experiments were conducted to test a set of general-purpose gamut-mapping functions. These gamut-mapping algorithms utilized contrast-preserving scaling functions. The algorithms were tested against the GCUSP gamut-mapping algorithm developed by Morovic and Luo (1998) which was shown to have very good universal gamut-mapping characteristics based on their experiments. The results of these experiments showed that vast improvements were obtained when linear lightness and chroma rescaling functions were replaced with contrast-preserving lightness and chroma rescaling functions. For these experiments, the gamut mapping consisted of sigmoidal lightness-remapping functions (Braun and Fairchild (1999a,c), Section 4) followed by either "knee" chromatic-compression functions (Gentile et al. (1990), Montag and Fairchild (1997)) or "sigmoid-like" chromatic-compression functions (Braun and Fairchild (1999b)).

5.2 CRT-to-Print Experiment

5.2.1 Summary

A psychophysical evaluation was performed to test the quality of several color gamut-mapping algorithms. The task was to determine which mapping strategy produced the
best matches to the original image. Observer preference was not considered. The algorithms consisted of both device-dependent and image-dependent mappings. Three types of lightness scaling functions (linear compression, chroma-weighted linear compression, and image-dependent sigmoidal compression) and four types of chromatic mapping functions were tested (linear compression, knee-point compression, "sigmoid-like" compression, and clipping). The source and destination devices considered were a monitor and a plain-paper inkjet printer respectively. The printer images were simulated on a monitor. The results showed that, for all the images tested, the algorithms that used image-dependent sigmoidal-lightness remapping functions produced superior matches to those that utilized linear lightness scaling. In addition, the results support using chromatic compression functions that are closely related to chromatic clipping functions.

5.2.2 Algorithms

A series of gamut-mapping algorithms were generated using the lightness and chroma compression schemes outlined in Sections 1.7, 1.8 and 3. These algorithms were grouped into three lightness rescaling categories:

1.) linear-lightness compression (LIN),

2.) chroma-weighted linear-lightness compression (GCUSP), and

3.) image-dependent sigmoidal-lightness compression (SIG))
Five chromatic scaling categories were used, and are shown in Figure 5-1:

1.) linear cusp-point scaling (LIN),

2.) knee-function cusp-point scaling (KNEE),

3.) cusp-point clipping (CLP),

4.) image-gamut based knee-function scaling (IMGGAM) (Note: The image gamut calculations are shown in Appendix B), and

5.) "sigmoid-like" cusp-point scaling (ENHANCE)).

Using various combinations of these lightness and chroma scaling categories, six hue-preserving, cusp-point based gamut-mapping strategies were developed and are shown in Table 5-1. These combinations were selected to illustrate the differences between linear and nonlinear gamut compression as well as to test the performance of new algorithms that were believed to have good general performance characteristics.
Figure 5-1 Illustration of the chromatic-compression functions listed in Table 5-1.
Table 5-1 Description of the gamut-mapping algorithms tested throughout this study.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Lightness Compression</th>
<th>Chroma Compression</th>
<th>Image- or Device-Depen. (Lightness /Chroma)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIN_LIN</td>
<td>Linear</td>
<td>Linear</td>
<td>Dev./Dev.</td>
</tr>
<tr>
<td>GCUSP</td>
<td>Chroma-Weighted Linear</td>
<td>Linear</td>
<td>Dev./Dev.</td>
</tr>
<tr>
<td>SIG_LIN</td>
<td>Sigmoidal</td>
<td>Linear</td>
<td>Img./Dev.</td>
</tr>
<tr>
<td>SIG_KNEE</td>
<td>Sigmoidal</td>
<td>Knee (90%)</td>
<td>Img./Dev.</td>
</tr>
<tr>
<td>SIG_CLP</td>
<td>Sigmoidal</td>
<td>Clipping</td>
<td>Img./Dev.</td>
</tr>
<tr>
<td>SIG_ENHANCE</td>
<td>Sigmoidal</td>
<td>3-Piece Linear (sigmoid-like)</td>
<td>Img./Dev.</td>
</tr>
<tr>
<td>SIG_IMGGAM</td>
<td>Sigmoidal</td>
<td>Knee (90%)</td>
<td>Img./Img.</td>
</tr>
</tbody>
</table>

5.2.3 Gamuts

The device gamuts were generated using the mountain-range representation described in Section 2 (Braun and Fairchild (1997)). The monitor gamut was generated using a gamut surface sampling of 20x20 points per face. These data were generated by processing the RGB digital counts through a GOG monitor model to get CIELAB points. The data were then transformed to the Hung and Berns hue-corrected CIELAB space using the LUTs generated in Section 3 (Braun, Ebner and Fairchild (1998)). The gamut surface points for the inkjet printer were generated by printing the RGB digital counts corresponding to the 20x20 grid per gamut face used for the monitor gamut specification. The spectral reflectance of these printed patches were measured using a Gretag SPM60 spectrophotometer. These spectral reflectances were converted to tristimulus values using the 10nm ASTM D65 weights for the 1931 standard observer. They were then converted
to CIELAB values using the paper white as the normalization factor. Finally, they were converted to the Hung and Berns hue-corrected CIELAB space. The resultant gamuts for the monitor and the printer are shown in Figure 5-2 and Figure 5-3. A difference of these gamuts is shown in Figure 5-4.

![Figure 5-2](image1.png)

*Figure 5-2. Mountain-range representation of the monitor gamut used in the monitor-to-print experiment.*

![Figure 5-3](image2.png)

*Figure 5-3. Mountain-range representation of the HP87Cxi inkjet printer gamut used in the monitor-to-print experiment.*
Figure 5-4. Contour plot of the difference between the monitor and the HP870Cxi inkjet printer gamuts used in the monitor-to-print experiment

5.2.4 Psychophysical Testing

For this experiment, seven pictorial images were used. Thumbnails of these images are shown in Appendix L. These images contained a wide variety of scene content that included memory colors such as skin-tones, sky, and grass. In addition, the features present in these images robustly spanned the monitor gamut. Special attention was given
to select images with regions of high chroma red, yellow, green, and blue. These colors are particularly affected when gamut mapping from monitor to print, since the monitor gamut is significantly larger than the printer gamut, as shown in Figure 5-4.

All the original images were from the full monitor gamut. These images were gamut mapped into the inkjet-printer gamut using the algorithms listed in Table 5-1. Instead of printing the CIELAB values of the gamut-mapped images, they were converted to monitor (RGB) digital counts using a GOG model for the monitor. This was done to control the viewing conditions. (The goal of this experiment was to test the gamut-mapping algorithms, not color appearance transformations.) If the reproductions were viewed as physical prints, there would have been mode-of-viewing, gloss, granularity, chromatic adaptation, and metamerism issues that would have affected the quality of the matches between the original and reproductions. They were eliminated from the experiment by viewing the gamut-constrained reproductions on the monitor. This helped to insure that the differences between the originals and reproductions were solely due to the gamut-mapping process and not some external factors.

Twenty observers performed the visual experiment which consisted of simultaneously viewing the original images and pairs of the gamut mapped reproductions, Figure 5-5. The observers were instructed to select the reproduction that was the closest match to the original. They were encouraged to keep in mind that the image they preferred may not be
the best match. Preference was considered in the experiments on gamut expansion, Section 6.

![Image of interface](image.png)

**Figure 5-5.** Illustration of the interface used to present the images in the monitor-to-print experiment. Three images were presented on a neutral gray background. The original image was the full gamut monitor original and the reproductions (A&B) were gamut-mapped versions. The observers viewed the images and picked the reproduction that was the best match to the original. They selected the winner by clicking on the button below the reproduction. All images were displayed on a monitor.

### 5.2.5 CRT-to-Print Results

A series of interval scales, shown in Figure 5-6, were developed that defined both the rank ordering of the algorithms' performance and a gauge of the relative difference among the techniques. These scales were generated using Thurstone's "Law of Comparative Judgments" (Torgerson (1967)). Incomplete matrix calculations were
applied to cases of unanimous agreement among observers. In those cases, it is impossible to directly calculate the Z-scores. The error bars shown on these plots represent the visual uncertainty between the algorithms. If the mean Z-score of an algorithm is contained within the error bars of another algorithm, the two algorithms have statistically the same visual performance. The confidence intervals used in the error bar calculations were derived from \( C = \frac{1.386}{\sqrt{N}} \), where N equals the number of observers.

Evaluation of the interval scales indicated that, across the images, the algorithms could be grouped into three significantly different categories. The first category of algorithms was the device-dependent linear lightness and linear chroma compression. This category included the GCUSP_LIN and the LIN_LIN algorithms. For all the images, these algorithms had much lower scale values than the images mapped using the sigmoidal lightness functions. This was primarily due to their low contrast which resulted from the linear dynamic range mapping.

The second category of results consisted of those images that were mapped using the sigmoidal lightness remapping functions and the linear chroma compression. This gamut mapping strategy created significantly better matches than the first category. This result stresses the importance of faithful reproduction of the lightness contrast of the scene, which is not found with straight linear lightness reproduction.
The third category of algorithms consisted of those that utilized both the sigmoidal lightness remapping functions and the non-linear chroma compression functions (SIG_KNEE, SIG_CLP, SIG_IMGGAM, SIG_ENHANCE). For all the images, these techniques produced significantly better matches than those produced by the first and second categories of algorithms. There were no significant differences noticed among these four algorithms. These gamut mapping routines resulted in very similar images since the knee-point of the mappings was set at 90-percent of the input gamut range (very similar to cusp-point clipping). The knee-point was set at the 90-percent point of the destination gamut based on the good performance of the clipping algorithms shown by Montag and Fairchild (1998) and because of the added flexibility to reduce the possible quantization artifacts of clipping.
Figure 5-6. Interval scale results for the monitor-to-print experiment. These scales are average across the seven scenes. Individual scales for the images are given in Appendix W.

5.2.6 Conclusions

The results of this study indicate that, for color gamut mapping of pictorial images, the biggest factor that affects the match between an original and a reproduction is the lightness-contrast rendition. This was shown by the significant improvements obtained using the image-dependent sigmoidal-lightness rescaling functions compared to the linear functions. Once the lightness contrast was appropriately mapped, the chromatic compression functions using non-linear knee-functions produced significantly superior
reproductions than the linear chromatic compression functions. Little difference was noticed among the chromatic-compression functions that were based on the image-gamut and those based on the device-gamut mismatches because of where the knee point was set. Thus, under general conditions it seems reasonable to forgo the complex image-gamut calculations for the chromatic compression when using scaling functions that are very similar to clipping.

5.3 Print-to-Print Experiment: Pilot

5.3.1 Summary

This experiment was designed as a pilot experiment to the print-to-print experiment detailed in Section 5.4. For this experiment, physical prints were utilized instead of simulating the prints on a monitor display, as was done in Section 5.2. Despite some sample generation and color management issues, the results of this experiment showed that sigmoidal-lightness rescaling functions and knee-function chromatic-scaling functions far out-perform gamut-mapping algorithms that use linear lightness and chromatic scaling. These results are directly in line with those from the monitor-to-print experiment, Section 5.2.
5.3.2 Goals

The main purpose of this experiment was to test the gamut-mapping strategies detailed in the monitor-to-print experiment under a different set of viewing and media conditions. The purpose was to determine if the performance of the gamut-mapping algorithms changed based on the image media. To achieve this, a print-to-print reproduction experiment was performed.

5.3.3 Gamuts

The output devices were the Xerox Regal MajestiK continuous-tone electrophotographic printer and the Xerox Xpress large format inkjet printer, both on plain paper. These printers represent typical graphic arts printers. The Regal MajestiK printer has a lightness dynamic range of approximately 15 to 100 CIELAB L* units, and the Xpress printer has a lightness dynamic range of approximately 22 to 100 CIELAB L* units. The dynamic range of the Fujix Pictrography 3000 printer used to generate the hardcopy originals is approximately 6 to 100 CIELAB L* units using glossy paper. (Note: The colorimetry was normalized to the luminance of the paper white.)

5.3.3.1 Destination Gamuts

Specification of the destination gamuts was relatively straight-forward. For each of the two output devices, a 3-dimensional LUT was supplied from Xerox that converted from a device CMY cube to CIELAB. These LUTs contained the CIELAB values for equally
spaced CMY grid points from a 10x10x10 lattice. In order to specify the gamut surface for these devices, a 20x20 grid of surface points from each face from a CMY cube was converted to CIELAB values using a tetrahedral-interpolation process defined by Hung (1993). These data were converted into Hung and Berns hue-corrected CIELAB color space. Mountain-range gamuts were created using the process defined in Section 2. These gamuts are shown in Figure 5-7 and Figure 5-8.

*Figure 5-7. Mountain-range representation of the Xerox Regal MajestiK gamut.*
5.3.3.2 Fujix Pictrography 3000 Gamut Estimation and Scanner Calibration

The gamut specification for the Fujix Pictrography 3000 printer, used to generate the originals, consisted of scanning prints of the gamut-surface points described in the previous section. In order to convert the scanned RGB digital counts to CIELAB, a colorimetric characterization was conducted for the Linotype-Hell Ultra-Saphir scanner. (Note: A description of the colorimetric-characterization process and the associated errors are given in Appendix M.) The resulting CIELAB values of the gamut-surface points were converted to Hung and Berns hue-corrected CIELAB values and used to create a mountain-range gamut. The Fujix gamut is shown in Figure 5-9.

*Figure 5-8. Mountain-range representation of the Xerox Xpress gamut.*
Figure 5-9. Mountain-range representation of the Fuji gamut generated from the scanner model.

5.3.3.3 Gamut Comparisons

The contour plots shown in Figure 5-10 and Figure 5-11 illustrate the differences between the Fujix gamut and the two destination gamuts. For the MajestiK gamut, Figure 5-10, the differences are mainly in the location of the yellow peak and the extra green and blue gamut found in the Fujix gamut. The Fujix gamut is larger than the Xpress gamut in nearly all regions of color space, Figure 5-11.
Figure 5-10. Contour plot of the difference between the Fujix gamut and the MajestiK gamut.
Figure 5-11. Contour plot of the differences between the Fujix gamut and the Xpress gamut.

5.3.4 Algorithms

The algorithms used in this experiment were the essentially the same as those used in the monitor-to-print experiment detailed in Section 5.2.2. The SIG_CLP algorithm was eliminated from this experiment, due to its similar characteristics and performance to the SIG_KNEE algorithm, found in the monitor-to-printer experiment.
Table 5-2. Description of the gamut-mapping algorithms tested in the print-to-print experiment.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Lightness Compression</th>
<th>Chroma Compression</th>
<th>Image- or Device-Depen. (Lightness/Chroma)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIN_LIN</td>
<td>Linear</td>
<td>Linear</td>
<td>Dev./Dev.</td>
</tr>
<tr>
<td>GCUSP</td>
<td>Chroma-Weighted Linear Linear</td>
<td>Dev./Dev.</td>
<td></td>
</tr>
<tr>
<td>SIG_LIN</td>
<td>Sigmoidal</td>
<td>Linear</td>
<td>Img./Dev.</td>
</tr>
<tr>
<td>SIG_KNEE</td>
<td>Sigmoidal</td>
<td>Knee (90%)</td>
<td>Img./Dev.</td>
</tr>
<tr>
<td>SIG_ENHANCE</td>
<td>Sigmoidal</td>
<td>3-Piece Linear (sigmoid-like) Img./Dev.</td>
<td></td>
</tr>
<tr>
<td>SIG_IMGGAM</td>
<td>Sigmoidal</td>
<td>Knee (90%)</td>
<td>Img./Img.</td>
</tr>
</tbody>
</table>

5.3.5 Color Management and Sample Preparation

The difficulty in performing this experiment were the color management and sample preparation issues that go into generating prints. The color management path had two parts: an input and an output path, Figure 5-12. The input path consisted of printing original images on the Fujix Pictrography 3000 printer and scanning them to get originals. The images were printed at 264 dpi and were scanned at 200 dpi. The scanned RGB digital counts were processed through the scanner characterization, given in Appendix M, to get CIELAB values in a similar manner as was used to generate the gamut surface of the Fujix printer.

The output path consisted of gamut mapping the scanned original files to each of the destination printers using the six gamut-mapping algorithms given in Table 5-2. The resultant CIELAB data were converted to 8-bit CIELAB TIFF files and sent to Xerox for printing. These CIELAB values were prepared for printing at 200 dpi. The MajestiK
printer prints at 400 dpi. The image data was not scaled during printing. Therefore, all of these reproductions were half the size of the originals. The Xpress printer scaled the 200 dpi data to fit a default page size that was slightly larger than would have been expected for a 200 dpi printer (i.e., it printed with an effective dpi of approximately 180). It was decided at the time that the scaled images would be acceptable for the experiment. The effects of the gamut mapping were evident despite the size differences.

Figure 5-12. Illustration of the color management path used to create the hardcopy reproductions for the print-to-print pilot experiment.

The Xerox tools used to perform the colorimetric characterization and calibration of the Xpress and the MajestiK printers were not capable of using the spectral power distribution of the actual light source for the viewing room where the experiment was
performed. Therefore, Illuminant D50 was used as an approximation. As such, illuminant metamericism affected the accuracy of the matches between the original Fujix Prints and the print reproductions. The metamericism resulted in a slight reddish color shift in the reproductions. In general, this did not affect the judgement among the gamut-mapping algorithms. The metamericism effect was small and constant across the different mapping algorithms. A detailed evaluation of the effects of the illuminant metamericism associated with these calibrations is given in Appendix N.

5.3.6 Psychophysical Testing

For the print-to-print experiment, four original printed images were gamut mapped into the two destination-printer gamuts given in Section 5.3.3.1. The four images selected for this experiment were different from those used in the monitor-to-printer experiment. Thumbnails of these images are shown in Appendix O. The images were viewed under a fluorescent D50 source (i.e., Greytag/Macbeth F40T12/75 and F40T12/50 fluorescent tubes). The observers' task was to rank the reproductions with respect to how well they matched the original image. The pilot experiment included 10 observers. Preference was not considered in these experiments.

A ranking experiment was performed because of the complexities of presenting printed images in a paired-comparison experiment. In addition, it is very easy for the observers to identify physical defects, such as scratches, wrinkles, creases, dents, etc. on the samples
that they can use to remember images from one trial to another. These visual cues would reduce the objectivity of the paired-comparison experiment.

As with the previous monitor-to-printer simulation experiment, the viewing conditions between the original and the reproduction were made as similar as possible, in this case by using prints for the original and reproductions. However, gloss, granularity, and resolution differences could not be fully eliminated because the media types used to generate the prints were different. For this experiment, these types of differences did not mask the differences attributed to the gamut-mapping algorithms.

5.3.7 Print-to-Print Results

The rank data associated with the print-to-print comparisons were converted to interval scale data, shown in Figure 5-13, using the "Comparative-Judgement Method of Data Reduction" (Bartleson and Grum (1984)). The interval scale values for each image are given in Appendix P. The error bars shown on these plots represent the visual uncertainty among the algorithms. If the mean Z-score of an algorithm is contained within the error bars of another algorithm, the two algorithms have statistically the same visual performance. The confidence intervals, C, shown as error bars in the figures were calculated by $C = \frac{1.386}{\sqrt{N}}$, where $N=21$ equaled the number of observers for the given experiment.
Figure 5-13. Interval scale results from the print-to-print pilot experiment. These scales are averaged over the four scenes.

The results from this experiment were essentially identical to those obtained from the monitor-to-print experiment. This provides good evidence that the characteristics of the gamut-mapping algorithms transfer from one mode of viewing to another. As with the monitor-to-print experiment, the results can be grouped into three categories. The first category contained the GCUSP and the LIN_LIN algorithms. They produced reproductions that were low in lightness contrast and chromatic content. The second category contained the SIG_LIN algorithm. The sigmoidal lightness rescaling was chosen as producing better matches than those that used linear lightness scaling. This emphasizes the importance that the lightness-tone reproduction has on the quality of a reproduction. The third category of mapping algorithms contained those that utilized the sigmoidal-
lightness rescaling followed by the nonlinear-chromatic compression (i.e., SIG_KNEE, SIG_IMGGAM, SIG_ENHANCE). These algorithms were all very similar based on having the knee point placed at 90-percent of the destination gamut range. The fact that they performed better than the algorithms in both Category 1 and Category 2 is evidence that the chromatic-compression function should keep the points well in gamut intact and only compress near and outside the gamut surface. The image-gamut compression performed the same as the device-gamut compressions because the compression functions looked very much like clipping. (Note: Recall, this was because the knee point was set at 90-percent of the destination gamut range. This essentially negates the usefulness of the image gamut calculations.)

5.3.8 Conclusions

The gamut-mapping approaches that utilized sigmoidal lightness mapping followed by knee-function chromatic compression, similar to cusp-point clipping, performed best over the various gamut-mapping cases studied. These algorithms had general success due in large part to the tone-preserving nature of the sigmoidal lightness remapping functions. In addition, performing the chromatic compression using scaling functions that maintain chromatic contrast was highly beneficial compared to linear chromatic compression.
5.4 Print-to-Print Experiment: Soft-proof

5.4.1 Goals

The goals of the second print-to-print experiment were to provide more data than was presented in the pilot experiment performed in Section 5.3. For this study, nine images were gamut mapped through a subset of the algorithms used in the previous two gamut-mapping experiments. A subset was used so that more images could be evaluated. The number of algorithms was reduced to limit the number of prints needed to be printed at Xerox. Based on the limitations associated with using the scanner to generate accurate colorimetric originals, a characterization was performed for the Fujix Pictrography 3000 printer used to create the print originals.

5.4.2 Characterization of the Fujix Pictrography 3000 Printer

Instead of using the scanner calibration to determine the CIELAB values of the original print files, a characterization was made of the Fujix Pictrography 3000 printer. This characterization consisted of printing a 10x10x10 lattice of uniformly space RGB values and measuring their spectral reflectance factors. These reflectance factors were then converted to CIELAB using the spectral power distribution of the print room and the color matching functions from the CIE 1931 Standard Observer. A 3-dimensional LUT was constructed that defined the transformation from RGB digital counts to CIELAB. The colorimetric accuracy of this transformation is detailed in Appendix Q and can be
summarized by a mean $\Delta E_{ab}^* = 2.1$ with a maximum $\Delta E_{ab}^* = 3.9$. These results were much better than those obtained for the scanner calibration and are within a visual tolerance for pictorial images (Stokes, Fairchild, and Berns (1992)).

5.4.3 Gamuts

All of the devices used in this study were the same as those used in the pilot experiment. The only difference was that they were recalibrated since the pilot experiment. Therefore, gamuts for the Xpress and the MajestiK printers were generated using updated device characterization LUTs supplied by Xerox in the same manner as described in Section 5.3.3. The gamut for the Fujix Pictrography 3000 printer was generated using its device characterization LUT in the same manner as the Xpress and MajestiK printers. These gamuts are shown in Figure 5-14 - Figure 5-16. Contour plot comparisons of the output gamut and the Fujix gamut are given in Figure 5-17 and Figure 5-18.
Figure 5-14. Mountain-range representation of the Xerox Royal MajestiK printer gamut.

Figure 5-15. Mountain-range representation of the Xerox Xpress printer gamut.
Figure 5-16. Mountain-range representation of the Fujix printer gamut generated from the printer characterization.
Figure 5-17. Contour plot of the difference between the Fujix gamut and the MajestiK gamut.
5.4.4 Algorithms

The algorithms selected for this experiment consisted of a subset of those used in the pilot experiment. The LIN_LIN algorithm was not used based on poor performance. The SIG_IMGGAM was not used based on similar performance to simpler algorithms. By eliminating these algorithms, more samples were able to be printed. Testing a greater
number of images was more important than testing these particular algorithms. The algorithms used in this experiment are summarized in Table 5-1.

Table 5-3. Description of the gamut-mapping algorithms tested in the print-to-print experiment.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Lightness Compression</th>
<th>Chroma Compression</th>
<th>Image- or Device-Depen. (Lightness /Chroma)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCUSP</td>
<td>Chroma-Weighted Linear</td>
<td>Linear</td>
<td>Dev./Dev.</td>
</tr>
<tr>
<td>SIG_LIN</td>
<td>Sigmoidal</td>
<td>Linear</td>
<td>Img./Dev.</td>
</tr>
<tr>
<td>SIG_KNEE</td>
<td>Sigmoidal</td>
<td>Knee (90%)</td>
<td>Img./Dev.</td>
</tr>
<tr>
<td>SIG_ENHANCE</td>
<td>Sigmoidal</td>
<td>3-Piece Linear (sigmoid-like)</td>
<td>Img./Dev.</td>
</tr>
<tr>
<td>GENERIC*</td>
<td>Sigmoidal</td>
<td>Knee (90%)</td>
<td>Fixed/Dev</td>
</tr>
</tbody>
</table>

* Details of the GENERIC algorithm are given in Section 5.4.6

5.4.5 Color Management

As with the pilot experiment the color management process consisted of two paths: input and output. The input path consisted of first printing the images on the Fujix Printer and then processing their RGB digital counts through the printer characterization model to get CIELAB values. The output path was the same as used in the pilot experiment.

The nine scenes were processed through the color management path for each of the five gamut-mapping algorithms. The file sizes were carefully controlled so that the prints were the same size as the original Fujix prints. (Note: The 264 dpi gamut-mapped images
were converted to 400 dpi and 720 dpi for the MajestiK and the Xpress printers respectively, using the bi-cubic interpolation functions in Photoshop 5.)

As with the previous printer calibrations, they were characterized for Illuminant D50. In this case the prints appeared to suffer more from illuminant metamerism than in the pilot experiment. The metamerism effects could be reduced by viewing the prints under a D65 daylight simulator, a filtered tungsten source, that is spectrally more similar to Illuminant D50 than the D50 fluorescent simulators used in the viewing room. This verified that the color shifts in the images were a result of the light source metamerism.

In addition to the metamerism effects, there were printer-quantization artifacts in the prints. Therefore, performing a visual experiment with these prints would not have fairly evaluated the gamut-mapping algorithms. As such, a softproofing experiment was performed (i.e., both the original and the reproductions were simulated on a monitor). The Xpress gamut essentially fit within a monitor gamut. As such, only a very small amount of clipping was required, for a few regions in color space, to fit the entire Xpress gamut inside the monitor gamut.

The MajestiK gamut was larger than the Xpress gamut and required more clipping to bring it within the monitor gamut. However, the MajestiK images were not used in the softproofing experiment. The differences between the MajestiK gamut and the Fujix gamut were minimal, Figure 5-17. The only region where the Fujix gamut was
significantly larger than the MajestiK gamut was in the blue. Of the nine scenes selected for this experiment, none had significant amounts of data in the region where these gamuts were dramatically different. As such, the gamut-mapped reproduction were very similar to the originals. The only differences were due to the lightness difference between the black points. (Note: The black-point difference between these devices was approximately 10 L* units.)

Softproofs of the reproduction were made by taking the 8-bit CIELAB images that were printed and converting them to monitor RGBs using a GOG model. In order to insure that all of the color values were within the monitor gamut a cusp-point clipping process was applied to all of the data that were outside the monitor gamut. Softproofs of the original Fujix prints were made by gamut mapping the CIELAB values, which resulted from processing the original RGBs through the printer characterization LUT, into the monitor gamut using cusp-point clipping. The accuracy of the gamut-mapped originals was visually evaluated. The decision was made that no real differences could be noticed between the Fujix prints and the Fujix softproofs.

5.4.6 Psychophysical Testing

The softproofs of the nine images and the four gamut-mapping algorithms were used in a paired-comparison experiment on a calibrated monitor set to have white-point chromaticities near D65. The images consisted of both standard portrait and landscape
scenes, as well as several more artistic images. For these scenes, there were fewer memory features for the observers to base their decisions. Thumbnails of these images are given in Appendix R. Twenty-one observers took part in this experiment. To be consistent with the print-experiment, the 264 dpi softproofs were converted to 100 dpi images. This made the displayed image the same size as the prints. The observers task was the same as that given in the monitor-to-print experiment: select the reproduction that was the best match to the original. Again the observers were encouraged to look at the entire scene before they made their judgement. They were instructed not to make their decisions based on preference.

One additional algorithm was added to this experiment. The algorithm denoted by GENERIC consisted of an image-independent sigmoidal lightness remapping followed by hue-preserving cusp-point knee scaling. This algorithm was, essentially identical to the SIG_KNEE algorithm except that the lightness scaling function was the same for all of the images. The form of the GENERIC lightness scaling was taken from the normal-lightness class given in Section 4 (Braun and Fairchild 1999a,c) This algorithm was added to test the utility of performing sigmoidal-lightness scaling using an image-independent approach that could be used in a color-management process like ICC.
algorithm performance could be generalized by three categories. (Note: While this experiment was performed using softproofs because of the sample generation issues previously stated, observations of the prints was consistent with the softproofing trends.) Category 1 contained the GCUSP algorithm (i.e., linear lightness and chroma compression). Category 2 contained the SIG_LIN algorithm. Again the significant difference obtained between the SIG_LIN algorithm and the GCUSP algorithm reconfirmed the role that the lightness contrast plays in the quality of the reproduction.

The final category (Category 3) contained the SIG_KNEE, SIG_ENHANCE, and the GENERIC algorithms. All these algorithms used chromatic compression functions that were nonlinear. They all compressed more near the edge of the gamut than in the central portion of the gamut. One of the interesting results of this experiment was that the GENERIC algorithm performed as well, on average, as the image-dependent SIG_KNEE and ENHANCE algorithms. This suggests that it would be possible to create a generic "profile" that could be used for all input images. The profile would be specific for a given destination dynamic range, but general for all input scenes. This would be very useful for implementation of these gamut-mapping algorithms in the framework of ICC color management. However, under extreme image histogram conditions it may still be more beneficial to use the image-dependent form of the sigmoidal-lightness remapping functions since they perform a tailored amount of compression in the highlight and shadowed regions in the scene.
5.4.8 Conclusions

As with the previous two experiments, the gamut-mapping approaches that utilized sigmoidal lightness mapping followed by knee function chromatic compression, similar to cusp-point clipping, performed best over the various gamut-mapping cases studied. These algorithms had general success due in large part to the tone-preserving nature of the sigmoidal-lightness remapping functions. In addition, performing the chromatic compression using scaling functions that maintain chromatic contrast was highly beneficial compared to linear-chromatic compression. Evidence was given for using a generic gamut-mapping algorithm that could be encoded into a profile like those used in ICC color management.

This experiment also highlights the importance of having a well color managed imaging system. Real display issues such as illuminant metamerism and halftone quantization can undo the efforts of carefully controlling the gamut mapping and the appearance modeling.

5.5 Recommendation of General Gamut-Mapping Algorithm

The recommendation for a General Gamut-Mapping Algorithm is to use the SIG_KNEE function used in the three gamut-mapping experiments. This algorithm possesses three key-components need to insure good performance:
1.) the use of the sigmoidal-lightness remapping function to circumvent the contrast lost in the lightness compression that is necessitated in nearly all gamut compression scenarios.

2.) the use of chromatic scaling functions that compress more near the edge of the gamut and compress toward the cusp point was shown to have significantly higher performance in these studies as well as others (Montag and Fairchild (1997), Gentile et al. (1990)). The use of a knee-compression function rather than a clipping function helps to maintain the texture of the high chroma features that could potentially be lost in clipping.

3.) by performing the compression in a hue-linearized color space, it is possible to maintain the hue of the original scene which is not always true when using CIELAB.
6 Gamut Expansion Experiment

6.1 Summary

The goals of this experiment were to determine if a gamut-mapping scheme that utilized gamut compression and expansion generated preferred reproductions to one that only performed gamut compression. In addition, this experiment tested the hypothesis that the preferred gamut expansion function would utilize the inverse of the best chromatic-compression function found in Section 5. This chromatic expansion function was tested using linear chromatic expansion. Linear lightness expansion was used to realize the extra contrast of destination gamut. While linear-lightness compression globally reduces lightness contrast, linear-lightness expansion globally increases the lightness contrast. The results of the experiments showed that using the extra lightness and chromatic dynamic range of the output device made enhancements to the reproductions that the observers preferred. Therefore, a gamut-mapping process that uses gamut expansion and compression should be preferred to just performing gamut compression.

In addition to trying the gamut-expansion case with knee-function chromatic expansion and linear-lightness expansion, nine other cases were considered. These cases utilized combinations of linear-chromatic expansion, knee-function chromatic expansion (with knee points at 50- and 90-percent), linear-lightness expansion, and inverse-sigmoidal lightness expansion.
The results of the experiments showed that the expansion function that used linear-lightness and (90-percent) knee-function chromatic expansion was equally preferred to one that used inverse-sigmoidal lightness and (50-percent) knee-function chromatic expansion. While there were no statistical difference between these algorithms, evidence was presented to suggest that the second case would produce more robust gamut expansions than the first.

6.2 Gamut-Expansion Algorithm

The gamut-expansion algorithms used in this experiment were very similar to the inverse of the gamut-compression algorithms used in the previous gamut-compression experiments. All of the expansion algorithms had two parts that were implemented sequentially. The first part was lightness expansion. Presumably, if the destination device had a larger lightness range than the source device it would be desirable to utilize this extra dynamic range to make the reproduction look better. The second part was to apply chromatic enhancement in the regions of the color space where the destination gamut exceeded the source gamut. The gamut-expansion regions were referred to as the TYPE II regions by Gentile et al. (1990), Figure 6-1. In their studies they only considered the TYPE I problem of gamut compression.
6.2.1 Lightness Expansion

For this experiment both linear and nonlinear lightness expansions were considered. Linear lightness expansion consisted of scaling the full source gamut lightness range into the full destination gamut lightness range using Equation 6-1:

\[
L^*_\text{out} = \left( \frac{L^*_{\text{src}} - L^*_{\text{src}(\text{min})}}{L^*_{\text{src}(\text{max})} - L^*_{\text{src}(\text{min})}} \right) (L^*_{\text{dest}(\text{max})} - L^*_{\text{dest}(\text{min})}) + L^*_{\text{dest}(\text{min})},
\]

(6-1)
where src and dest represent the source and destination gamuts respectively. The result of applying linear lightness expansion is that the perceived lightness contrast in the image increases and the mean lightness of the image decreases.

In order to counteract the darkening associated with the linear-lightness expansion, a nonlinear-lightness expansion function was considered. This function was based on the inverse of the sigmoidal-compression functions used in the previous gamut-mapping experiments. Recall that the lightness compression was a function of the lightness histogram of the source image and the dynamic range of the destination device. In the gamut expansion case, it was not possible to consider the form of the lightness histogram in the selection of the remapping function. The reason for this was that the histogram data used to populate the sigmoidal-contrast model (i.e., the gamut-compression case) were calculated for images that came from a lightness range of [0 100]. For lightness expansion, the input images came from a device with a reduced lightness range of [L^{*}_{min}, 100]. As such, their histograms were compressed and shifted into a smaller range. The consequence of this was that the histogram statistic for the 75-percent point of the cumulative histogram was shifted as well. Therefore, it was not possible to directly relate the histogram statistics from the full dynamic range case, (i.e., [0 100]), to the case where the input image histogram was constrained to a smaller lightness region.

Based on the restrictions listed above, the gamut-expansion model consisted of the inverse functions used for the normal-lightness class of the sigmoidal-contrast model.
This class was selected based on its good performance as a general gamut-compression function, shown in Section 5.4. Therefore, consider the example where the source device had a minimum lightness equal to \( L^*_{\text{src(min)}} \) and the destination device had a minimum lightness equal to \( L^*_{\text{dest(min)}} \), where \( L^*_{\text{dest(min)}} < L^*_{\text{src(min)}} \). The inverse-sigmoidal remapping function would then be obtained by inverting the remapping function obtained from the sigmoidal contrast model for a black point of \( L^*_{\text{src(min)}} \) and a 75-percent point of 51 lightness units. (Note: The 75-percent point of 51 lightness units corresponds to the histogram point for the normal lightness class.)

For the purposes of these experiments, the sigmoidal lightness-remapping functions were implemented as 12-bit look-up tables. Inversion of these tables is described in Appendix T.

### 6.2.2 Chromatic Expansion

In this experiment both linear and knee-function chromatic scaling functions were utilized. As with the gamut-compression functions described in Section 5 the scaling direction was toward the gamut cusp point for each individual hue angle. The chromatic scaling was implemented in a manner similar as described for the gamut compression algorithms (Section 5). In this case, the chromatic-expansion knee functions had an identity region and an expansion region, Figure 6-2.
Another algorithm included a modification to the knee-function scaling used in the gamut-compression experiments (i.e., knee point at 90-percent). For some of the cases tested the knee point was moved to 50-percent of the destination gamut range, Figure 6-2. The reason for this was that in preliminary testing, the gamut-expansion process occasionally produced contours in the image when the knee was set at 90-percent. This was most noticeable in features that contained chromatic sweeps from low to high chroma. By moving the knee point to 50-percent of the destination gamut range, the sweeps were smoother.

![Diagram](image.png)

*Figure 6-2. Illustration of the chromatic enhancement function used in gamut-expansion regions.*
In addition to varying the location of the knee point, the amount of chromatic enhancement was controlled, in some cases, by placing constraints on the maximum percentage of chromatic boost. This constraint process, shown in Figure 6-3, can help control how much extra chromatic range is added to a feature. This helps to insure that they are not boosted in chroma to the point where they appear unnatural. The example shown in Figure 6-3 constrains the boost to 30-percent. The shaded gamut is a scaled version of the destination gamut. The destination gamut is scaled down so that it is only 30-percent bigger than the source gamut.
Figure 6-3. Illustration of the gamut expansion constraints placed on the maximum chromatic boost. The shaded gamut is a scaled version of the destination gamut. The constrained gamut is only 30-percent larger than the source gamut.

6.2.3 Algorithms Tested

In this experiment, the twelve gamut-expansion cases were considered and they are listed in Table 6-1. Not all of the twelve cases were tested in the final psychophysical evaluation for several reasons. (The author pre-screened these algorithms by looking at 31 gamut-mapped scenes.) First, testing all twelve cases would have resulted in too many pairs for the observers to judge in one session. Second, for some of the algorithms, the unconstrained (FULL) and the constrained expansions resulted in the essentially the same...
gamut-mapped image. For example, suppose that the expansion constraint was set to 30-percent. Along any scaling line the chromatic data could only be boosted by a maximum of 30-percent of its input value. Depending on the region of color space, the ratio of the destination gamut to the source gamut may or may not have exceeded 30-percent. For those regions that did not exceed 30-percent, there was no difference between the unconstrained (FULL) and the constrained expansions, Figure 6-4.

![Diagram](image)

**Figure 6-4.** For this hue leaf, the chromatic boost does not exceed 30-percent for any region of the gamut. The algorithms that constrain the percent boost to 30-percent will produce the same image as those that use the FULL range of the destination gamut.
Table 6-1. Key to the gamut-expansion cases used in this study. The percentages shown in the “Chromatic Scaling Function” column identify where the knee point was set. The values shown in the “Percent Gamut Expansion” column represent the maximum amount of chromatic boost that was allowed. FULL indicates that the entire range of the destination gamut was used. A given percentage represents the constrained amount of boost that was allowed. The Y/N symbol indicates whether the case was tested in the psychophysical experiment.

<table>
<thead>
<tr>
<th>Case</th>
<th>Lightness Scaling Function</th>
<th>Chromatic Scaling Function</th>
<th>Percent Gamut Expansion</th>
<th>Tested</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Linear</td>
<td>Linear</td>
<td>FULL</td>
<td>N</td>
</tr>
<tr>
<td>2</td>
<td>Linear</td>
<td>Knee (90%)</td>
<td>FULL</td>
<td>N</td>
</tr>
<tr>
<td>3</td>
<td>Linear</td>
<td>-</td>
<td>FULL</td>
<td>Y</td>
</tr>
<tr>
<td>4</td>
<td>Linear</td>
<td>Linear</td>
<td>FULL</td>
<td>Y</td>
</tr>
<tr>
<td>5</td>
<td>Linear</td>
<td>Knee (90%)</td>
<td>FULL</td>
<td>Y</td>
</tr>
<tr>
<td>6</td>
<td>Linear</td>
<td>Knee (50%)</td>
<td>FULL</td>
<td>Y</td>
</tr>
<tr>
<td>7</td>
<td>Inverse Sigmoidal</td>
<td>Knee (50%)</td>
<td>FULL</td>
<td>Y</td>
</tr>
<tr>
<td>8</td>
<td>Inverse Sigmoidal</td>
<td>Knee (50%)</td>
<td>10%</td>
<td>N</td>
</tr>
<tr>
<td>9</td>
<td>Inverse Sigmoidal</td>
<td>Knee (50%)</td>
<td>20%</td>
<td>N</td>
</tr>
<tr>
<td>10</td>
<td>Inverse Sigmoidal</td>
<td>Knee (50%)</td>
<td>30%</td>
<td>Y</td>
</tr>
<tr>
<td>11</td>
<td>Inverse Sigmoidal</td>
<td>Knee (50%)</td>
<td>40%</td>
<td>N</td>
</tr>
<tr>
<td>12</td>
<td>Inverse Sigmoidal</td>
<td>Knee (50%)</td>
<td>50%</td>
<td>N</td>
</tr>
</tbody>
</table>

The first three gamut-expansion cases listed in Table 6-1 were designed to differentiate between chromatic and lightness expansion. In order to test chromatic expansion independently of lightness dynamic-range scaling, the algorithms tested in Case 1 and Case 2 did not apply lightness expansion. Case 3 applied lightness dynamic range stretching, but no chromatic stretching. All of the other cases performed both lightness and chromatic stretching.
The first two cases required that the destination gamut be constrained by clipping the bottom of the gamut at the lightness of the source gamut black point, Figure 6-5. If the destination gamut were not clipped at the source-gamut black-point lightness, then lightness expansion would occur for the low lightness input pixels. This gamut clipping process resulted in an artificially shaped destination gamut. The gamut clipping resulted in “blocking” in some of the low lightness features. For example, large numbers of pixels were mapped to the same lightness level at the bottom of the gamut. This is shown in Figure 6-5 by the “clipping line”. Many of the input pixels were mapped to this line. As a result, the shadowed and dark regions in the scenes had flat texture. These two cases did not result in realistic reproductions and were therefore eliminated from the psychophysical evaluations.
Figure 6-5. Illustration of the gamut clipping process that was required to insure that no lightness dynamic range expansion was performed for Case 1 and Case 2 given in Table 6-1. Note the clipping line at the bottom of the gamut. Pixels mapped along lines from the cusp point, that intersect this line, may have resulted in lightness clipping at the black point of the source device.

6.2.4 Psychophysical Testing

The gamut expansion experiment was performed from hardcopy-to-softcopy. The context of this experiment fits in with the world-wide-web example given in Section 1.10. The source device was the Fujix Pictrography 3000 printer. The destination gamut was a Sony GDM-2000TC graphic arts monitor adjusted to a D65 white point. A comparison of these
two gamuts is shown in Figure 6-6 and Figure 6-7. The lightness dynamic range difference between these two gamuts was approximately 6 CIELAB L* units.

Figure 6-6. Surface plot of the difference between the gamuts used in the gamut expansion experiments. The source gamut was the Fujix gamut and the destination gamut was the CRT. The negative regions correspond to where the Fujix gamut was larger than the monitor gamut.
Figure 6-7. Contour plot of the difference between the monitor gamut and the Fujix gamut used in the gamut-expansion experiment.

Eight of the initial 31 scenes were selected in the psychophysical experiment. Thumbnails of these images are shown in Appendix U. These scenes were selected because they had representative features that covered a wide range of gamut-expansion issues. At the onset of this experiment, it was believed that linear chromatic scaling would unnaturally enhance low chroma features like skin-tones and neutrals in the scene. One of the eight scenes contained skin tones. Additionally, it was speculated that using the full destination range might cause the reproductions to look unnaturally chromatic. Therefore, some scenes with high-chroma features were selected. This experiment was
designed to show the utility of gamut expansion for the hardcopy-to-monitor viewing. For this experiment the blue and green regions of color space had the most extra range for gamut expansion. (Note: These regions are represented by the light contours in Figure 6-7.) Therefore, images with data in these regions were selected so that the most dramatic effects of the gamut-expansion process could be examined.

A forced-choice paired-comparison experiment was performed using the eight scenes and the six algorithms highlighted in column four of Table 6-1. Twenty-two observers took part in the experiment. The observers were asked to pick the reproduction, either A or B, that they preferred. No original was present. The context of this task was that these reproductions represent two prints. The question asked to the observers was “Which would you rather have?” The observers were encouraged to consider all aspects of the scene in their judgment.

The Sony GDM-2000TC pre-press monitor used in this experiment was set to have white-point chromaticites similar to Illuminant D65. A colorimetric characterization of this monitor was then performed using the GOG process outlined by Berns, Motta, and Gorzynski (1993). These parameters were used to convert CIELAB values to monitor RGB digital counts. The processing path from input Fujix RGB digital counts to output monitor RGB digital counts is shown in Figure 6-8. This is intended to be representative of a reproduction chain where an input image is printed, scanned, gamut mapped, and displayed on a CRT.
6.3 Experimental Results

The results of the paired comparison experiment are summarized in the interval scale shown in Figure 6-9. The individual interval scales for the eight scenes are shown in Appendix V. Gamut-mapping schemes that utilized both compression and expansion were preferred over the one that performed only gamut compression. This indicates that on average it is a good idea to perform gamut expansion. When the results are averaged across the eight scenes, the algorithms that utilized the "knee-function" chromatic
expansion functions (Cases 5, 6, 7, and 10) were preferred. On average there were no significant differences obtained between:

- the knee function position at 50% (Case 6) and 90% (Case 5) of the distance to the destination gamut surface,

the linear (Case 6) and inverse-sigmoidal lightness expansion (Case 7), or

the constrained (Case 10) and unconstrained expansion (Case 7).

Figure 6-9. Interval scale values for the algorithms tested in the gamut-expansion experiment (averaged across the eight scenes). The confidence intervals given by the error bars were calculated as ± 1.386/sqrt(N*I) where N was the number of observers and I was the number of scenes (N=22, I=8).

The algorithm that used linear-lightness and linear-chroma scaling (Case 4) did not perform well. In general, this algorithm made the images look too chromatic and dark.
Similarly, the algorithm that just enhanced the lightness dynamic range (Case 3) did not perform well compared to those that performed chromatic enhancement as well as lightness expansion.

Further insight into the gamut-expansion process can be obtained by looking at the interval scales for the individual images. In the “RedBarn” image, the linear-lightness and linear-chromatic expansion algorithm (Case 4) performed significantly worse than the other algorithms. The reason for this was that the linear chromatic expansion away from the cusp point resulted in a mottling of some low chroma features in the clouds. This gave the clouds an unpleasant pattern that looked like film grain noise in an enlargement of low-resolution photographic film.

In the “Eiffel Tower” scene, there was an artifact in the sky when the knee function was set to 90-percent. This image had a lightness and chroma gradient that went from the bottom (high L*, low C*) to the top of the scene (low L*, high C*). Only the low lightness data were within the expansion regions, and were therefore boosted in chroma and reduced in lightness. (Note: They were reduced in lightness because they were expanded toward the bottom of the gamut.) This resulted in a rather sharp contour that broke up the smooth gradient in the sky. When the knee point was moved to 50-percent, this contour was reduced. Observers were relatively indifferent to or unaware of this artifact. However, the 50-percent knee point is recommended as it received the same rating but did not exhibit this artifact.
The “Daffodil” scene gives more evidence against using linear chromatic expansion, as evidenced by the poor performance of the Case 4 algorithm. In this scene, the linear chromatic expansion caused the “blue” sky to become cartoonishly chromatic. In addition to the overly chromatic sky, another artifact of linear-chromatic expansion was realized in the clouds. The “white” clouds were on a very blue background in the scene. The edges of the clouds were “fuzzy” compared to the main body of the cloud. This fuzziness resulted from some of the blue sky showing through the cloud. Since the edges of the clouds had a small degree of blue in them (i.e., some chroma) their chroma was boosted using the linear-chromatic expansion (Case 4). This gave them the appearance of a bright-blue halo. The halo was less severe in the cases where the knee function was set at 50-percent of the destination gamut range (Case 5). It was not present in the gamut expansion functions that used knee function expansion at 90-percent because most of the chromatic range was left unexpanded (Case 6).

A similar artifact was noticed for Case 4 in the “Hats” scene. The chroma of a region of clouds was increased to an extent that it appeared as though there was a bright green blob in the clouds, which observers found objectionable. As with the other images, those algorithms that used knee-function expansion did not contain this artifact.

The concern with using linear lightness expansion was that it decreases the mean lightness of the image, making the image appear darker. This problem was substantiated only in the “FamilyOnBlanket” scene. In this case, the face of the woman in the scene
was slightly shadowed. The linear lightness expansion caused the shadowed region of her face to appear too dark. The inverse-sigmoidal expansion function helped to reduce this effect.

The constraint placed on the maximum amount of chromatic boost (Case 10), was also beneficial for the “FamilyOnBlanket” scene. The destination gamut was much bigger than the source gamut in the blue region of color space. In the unconstrained case, the chroma of the man’s blue shirt was increased to the point of looking fluorescent. By constraining the chromatic boost to 30-percent or less, his shirt was reproduced more naturally than in the unconstrained case. The constraint of 30-percent chromatic boost did not limit the quality of the reproduction of his shirt. It was more chromatic and generally more preferable than with no boost. This is reflected in the interval scale value for the Case 10 algorithm for this scene.

### 6.4 Conclusions

The results of this experiment showed that the gamut-compressed/expanded reproductions were preferred over the gamut-compressed reproductions.

While the experimental results did not definitively show that constraining the maximum allowable boost in chroma was preferred, intuitively there should be an upper limit to how much added chroma the observers will prefer before the image looks unnatural. Therefore, it is recommended that gamut expansion be performed with a constraint on the
maximum chromatic boost. In these experiments, the Case 10 algorithm, which limited the boost to 30-percent, performed as well as the cases that used the full range.

The inverse-sigmoidal lightness expansion (Cases 7 and 10) performed as well as or better than linear lightness expansion (all other cases). This process can help avoid the darkening of the image that results from linear lightness expansion and is therefore recommended.

Moving the knee point from 90-percent to 50-percent will ultimately aid in the gamut expansion process. This will help to maintain the smoothness of chromatic sweeps. This is particularly important for the reproduction of skies. However, for compression the knee point should remain at the 90-percent point.

Overall, gamut mapping using a combination of gamut compression and gamut expansion will result in a more universally appealing reproduction than just applying gamut compression. The recommended attributes were all contained in the Case 10 algorithm, which performed well for most of the images tested. Expansion will be beneficial in treating the color management associated with reproducing SWOP-encoded files on printers with larger gamuts than offset lithography or when advertising on the world-wide web.
7 Incorporation of Gamut-Mapping Algorithms into Color-Management Systems

Two basic types of color-management systems can be generalized. The first uses a simple color-management engine and relies on device manufacturers to supply “smart” device profiles to control the color-management. The second uses a “smart” color-management engine and relies on the device manufacturers to supply profiles that simply characterize the device. In the first case, the gamut mapping is the responsibility of the device manufacturers and in the second case the gamut mapping is handled by the color-management engines. The following sections incorporate the general-purpose gamut-mapping algorithm defined in Section 5.5 into the computational paradigm of each engine.

Currently, the default paradigm specified by the International Color Consortium (ICC) is designed around using “smart” profiles with a comparatively simple color-management engine. While there are opportunities for individual designers of the color-management engines to add extra processing capabilities through third party color-management modules (CMMs), the baseline process uses a simple engine. Unfortunately, information regarding how to design and implement a color-management strategy using ICC profiles is elusive in the open literature.
7.1 "Smart" Color Profiles

In the first case, the color-management engine relies on the device profiles to contain all of the necessary information to perform the color-space conversions. For example, an input profile contains all of the necessary information to transform the device-dependent color coordinates into the device-independent color coordinates of the profile-connection space (PCS). Likewise, profiles associated with output devices need to have all of the necessary information to convert from the PCS to the device-dependent color coordinates of the output device (e.g., CMY, CMYK, RGB, etc.). The color-management engine simply processes image data, on a pixel-by-pixel basis, through the input and output profiles. These in-gamut device-dependent data are then supplied to the output device.

The general purpose gamut-mapping algorithm described in Section 5.5 requires that image-dependent data be generated to guide the gamut-mapping process. As such it is not possible to implement this algorithm using the "smart"-profile paradigm currently baseline by ICC profiles.

Even if the image-dependent nature of this algorithm was eliminated, it would be very cumbersome to implement this gamut-mapping process using ICC profiles. The form of the gamut-compression functions relies on the relative mismatch of the source and the destination gamuts. As such it would be impossible to generalize a single output profile that was tailored to all input gamuts. The current philosophy of the ICC is that different
rendering intents can give profile designers hints on the form of the input gamut. With this information, they can engineer a profile that will satisfactorily process images designed for that rendering intent.

In order to perform a gamut-mapping process that requires the exact forms of the input and output gamuts using a look-up table (LUT) process, a custom LUT would need to be generated for each input/output gamut pair. In the ICC format, this profile might take on the form of a device-link profile. Thus, input device data could be directly processed to output device data using a single LUT. This would be very difficult to implement because input and output devices are made by different manufacturers. By its nature, this process violates the spirit of the open ICC color-management environment and the profile-connection space.

### 7.2 "Smart" Color-Management Modules

By shifting the color management from the device profiles to the color-management engine, it is possible to utilize all of the gamut-mapping algorithms in this thesis without defining custom device profiles. In this case, the device profile is simply a characterization tool. The profile of an output device needs only to be defined for points within its color gamut. The gamut-mapping responsibility now lies with the color-management engine, not the profile manufacturer. Assuming that the application programmer's interface (API) allows the color-management engine to perform pre-
scanning to calculate image statistics such as histograms and moments, it would be possible to implement the algorithm defined in Section 5.5. (The current APIs used by ColorSync and ICM support buffering the entire image within the CMM but this is not the common implementation (Rosen (1999)).)

Consider the case where input from any system is to be reproduced on an output device. In the case of the "smart" color-management engine, the input profile converts image data to the PCS. If the input profile is associated with a display device such as a monitor, the color gamut of the source device can be generated by the color-management engine. If the input profile is associated with a scanning system, the source gamut either needs to be 1.) tagged with the scan, 2.) inferred by the color-management engine from the image data, or 3.) assumed to be the image data (e.g., using image-gamut process used in this study).

The output profile needs to define the transformation from the PCS to the device-dependent color-coordinates only for in-gamut colors. The color-management engine is required to apply all the required gamut-mapping transformations to insure the source data is within the destination gamut. The role of the color-management engine has been elevated from number-cruncher to image processor. This shift in paradigm from the simple to the "smart" color-management engine opens the possibilities for image-dependent gamut-mapping algorithms. It is hoped that the ICC will gravitate toward
using "smarter" color-management engines so that more sophisticated gamut-mapping algorithms can be implemented.

7.3 Hybrid Profiles and Custom CMM

The gamut-mapping algorithm given in Section 5.5. of this thesis could be implemented using the current ICC profile specification and a third-party CMM. The following example illustrates this process. This is not a reflection of how current color-management is performed; it is simply an illustration of how the system can be manipulated to perform the operations needed to implement image-dependent gamut mappings.

The output device profiles used in this process would utilize the private tags of the ICC format to store an inverse device characterization LUT that transforms in-gamut PCS data back into device-dependent coordinates. Another private tag would be used to imbed the device gamut specified using the process detail in Section 2.

The input device profile would contain the standard transformation to process data from device dependent space to the PCS as well as contain private tags that specify the gamut according to the process given in Section 2. This input profile would be tagged to the image that was being processed. By tagging the profile to the image, another custom profile tag could be used to store the lightness histogram of the input image. This histogram would be extracted from the profile by a customized CMM that performed the gamut mapping.
A customized CMM would read in the image with the custom tagged source profile and the output devices custom tagged profile. The custom CMM would then extract the source image’s lightness histogram from the private-tagged input profile and generate an image-dependent sigmoidal-lightness remapping function using the model given in Section 4. The CMM would then extract the source and destination gamuts from the input and output profiles private gamut tags. These gamuts would be used to generate a PCS-to-PCS space transformation using the chromatic-compression functions outlined in Section 5. The custom CMM would then concatenate the LUTs that processed from input-device coordinates to output-device coordinates to create a custom device-link profile for the particular input image. The image would be processed through this profile and rendered on the output device. This process would work within the current strategies outlined by the two major color-management systems (i.e., ColorSync and ICM).
8 Conclusions and Recommendations

The goal of this thesis was to develop a robust process for color-gamut mapping of pictorial images. In order to achieve this goal, a series of key components were developed:

1.) A general gamut specification process (Section 2) (Braun and Fairchild (1997)).

2.) A hue-linearized color space for preservation of perceived hue (Section 3) (Braun, Ebner, and Fairchild (1998)).

3.) A general contrast-preserving lightness-rescaling process, tuned to gamut dynamic ranges and image content (Section 4) (Braun and Fairchild (1999a,c)).

4.) An image-gamut based chromatic scaling process that performs similar to chromatic clipping without the drawbacks of clipping (Section 5) (Braun and Fairchild (1999b,d)).

5.) A gamut-expansion process that boosts the lightness contrast and the chromatic content of an image in a natural manner (Section 6).
Using these components, a series of gamut-mapping experiments were performed (Section 3, Section 5, and Section 6). The gamut-mapping experiment described in Section 3 showed that gamut mapping in a hue-linearized color space dramatically improved the accuracy of the hue reproduction compared to mapping in CIELAB. A recommendation for further research in this area is to test the IPT color space (Ebner and Fairchild (1998b)) in a gamut-mapping experiment. This color space could be very valuable if shown to perform better than the LUT-based approach utilized in this experiment.

The three gamut-mapping experiments performed in Section 5 showed that gamut-mapping algorithms utilizing contrast-preserving sigmoidal-lightness scaling functions produced significantly better gamut-mapped reproductions than algorithms that performed linear-lightness compression. In addition, these experiments reconfirmed that significant amounts of chromatic content can be maintained by using non-linear chromatic-scaling functions (such as knee-functions) rather than linear-chromatic scaling functions.

Based on the results of these psychophysical experiments, it was possible to generalize a gamut-mapping strategy that is robust for gamut mapping of pictorial images. This strategy uses an image-dependent, contrast-preserving, sigmoidal-lightness rescaling followed by knee-function chromatic scaling that acts similarly to chromatic clipping. For images from unknown source gamut, this process can utilize an image-gamut scaling for
added quality (Section 1.8.6). This algorithm performed in the top category for each of
the gamut-mapping experiments. This gamut mapping process is illustrated in Figures 8-
1 – 8-4.

In addition to gamut compression, a gamut-expansion function was given that was based
on the inverse of the general gamut-compression algorithm previously mentioned. This
process uses an inverse sigmoidal-lightness expansion function followed by compression
with a knee-function at 90-percent for Type I regions and expansion using an inverse
knee-function at 50-percent with a chromatic boost constraint of 30-percent for Type II
regions. This algorithm was shown to result in more preferred reproductions than using a
compression algorithm alone or one that used linear expansion. Thus, it is possible to
generalize a process for using the extra dynamic range of destination gamut that is not
contained in the source gamut.

A recommendation for further study in gamut mapping is to evaluate the use of
algorithms that relax the hue preservation constraint utilized by the algorithms tested in
this study. There are several difficulties that would need to be overcome for this to work.
The color space would need to be optimized such that equal steps in hue angle would
predict equal changes in perceived hue throughout the color space. This might be difficult
with the hue spacing of a color space such as CIELAB. Without more uniformity
between hue angles it would be difficult to generalize a solution that allowed hue to vary
for all input/output gamut combinations. Recently, this topic has been addressed by
Kuehni (1999a,b). In his research, Kuehni has developed a color space that is based on uniformly spacing the Munsell renotation data. One of the nice features of his color space is that greater uniformity exist between metric hue angles. This helps make it possible to vary metric hue angle in a more predictable manner than is afforded in the CIELAB color space. The main limitation to this color space, for gamut mapping, is that it has been designed around the Munsell renotation data that has a limited chromatic range. It would be interesting to combine Kuehni’s color space with the extended chromatic data used by Ebner and Fairchild’s (1998b) IPT color space.

Figure 8-1. High-level flow chart of the gamut-mapping process. Flow charts for the “Lightness Compression” and the “Chromatic Compression” pieces are given in Figures 8-2, 8-3 and 8-4.
Figure 8-2. Illustration of the lightness-mapping process used in the general-purpose gamut-mapping algorithm.
Figure 8-3. Illustration of the chromatic mapping stage of the general-purpose gamut-mapping algorithm.
LC coords. in Polar form about the Cusp Point \([p, \alpha]\)

CurrentAlpha = 0

Find all pixels in CurrentAlpha

Calculate Scale Factor for CurrentAlpha from Src. & Dest. LGB

Generate Knee-Function Based on CurrentAlpha Scale Factors

Process all \(p\)'s from CurrentAlpha Through Knee Function

Transform \([p, \alpha]\) from CurrentAlpha into LC coords.

IF CurrentAlpha \(\neq 0\)

Gamut Mapped LC coords. for CurrentHueAngle

Figure 8-4. Polar gamut-mapping process performed in the “Gamut Map” stage given in Figure 8-3.
9 References


Hains, C.M., Personal correspondence.


Rosen, M., Personal correspondence.


Appendix A - Homomorphic Filtering

The following is a derivation of Equation 1-8. This represents the extreme case where the width of the low-pass filter, GAUS(u,v), goes to a delta function. Only the DC signal is included in the low-frequency signal. The result is an enhancement that is a power function rescaling followed by a linear dynamic range compression. This form is shown by the following derivation:

\[ f(x,y) = i(x,y)r(x,y) \quad (A-1) \]
\[ z(x,y) = \ln(f(x,y)) = \ln(i(x,y)) + \ln(r(x,y)) \quad (A-2) \]
\[ Z(u,v) = \mathcal{Z}\{z(x,y)\} = I(u,v) + R(u,v) \quad (A-3) \]
\[ S(u,v) = H(u,v)Z(u,v) \quad (A-4) \]

let,
\[ H(u,v) = \beta - \gamma \delta(u,v) \quad (A-5) \]
\[ S(u,v) = \beta Z(u,v) - \gamma \delta(u,v)Z(u,v) \quad (A-6) \]
\[ S(u,v) = \beta Z(u,v) - \gamma Z(u = 0, v = 0) \quad (A-7) \]
\[ s(x,y) = \mathcal{Z}^{-1}\{S(u,v)\} \quad (A-8) \]
\[ s(x,y) = \beta \tilde{z}(x,y) - \gamma \tilde{Z}(u = 0, v = 0) \quad (A-9) \]
\[ s(x,y) = \beta \tilde{z}(x,y) - \gamma' \quad (A-10) \]
\[ g(x,y) = e^{s(x,y)} \quad (A-11) \]
\[ g(x,y) = e^{(\beta \tilde{z}(x,y) - \gamma')} = \left(e^{\tilde{z}(x,y)}\right)^{\beta} e^{-\gamma'} \quad (A-12) \]
\[ g(x,y) = \gamma' \tilde{f}(x,y)^{\beta} \quad (A-13) \]
Appendix B - Image-Gamut Calculation

The following is a list of the steps used to generate an image gamut. The image-gamut contour was set at 95-percent of the cumulative histogram. These calculations are performed in polar space about the cusp point.

1. Convert the CIELAB image data into CIELAB LCh coordinates.

2. Sort the image data into the nearest integer hue angle between 0 and 360 degrees.

3. For each hue angle convert the L* and the C_ab* values for the current hue and those of the two neighboring hue angles into polar coordinates about the cusp-point (L_{cusp}*) of the current hue center. This results in radius (\rho) and angle (\alpha) terms, for each point, given respectively by:

\[
\rho = \sqrt{(L^* - L_{cusp}^*)^2 + (C_{ab}^*)^2}
\]

\[
\alpha = \tan^{-1}\left(\frac{L^* - L_{cusp}^*}{C_{ab}^*}\right) \times \frac{180^\circ}{\pi} + 90^\circ
\]

4. Calculate the 2-dimensional [\rho, \alpha] histogram of the pixels in that hue and the two adjacent hue angles. The bin sizes used in the histogram calculation are [\Delta \rho, \Delta \alpha] = [1, 1]. The two adjacent hue angle bins are used in the histogram process to insure smoothness between the hue segments around the hue circle. The gives a 181\times N matrix of histogram values where the rows of the matrix are the \alpha values from [0, 180] degrees in 1
degree steps. The N columns of the matrix are in distance units from the
cusp-point and cover the range of [0, k] where k = nearest integer of
\(\text{max}(\rho)\) and N = k+1.

5. Calculate the cumulative histogram for each \(\alpha\) by taking the cumulative
sum along the rows of the \([\rho, \alpha]\) histogram.

6. Normalize each row of the cumulative histogram by dividing by the
maximum value in each row.

7. For each row of the normalized cumulative histogram, determine the
column (\(\rho\)) where the cumulative histogram equals 0.95. This radius is
used to represent the image gamut for the current hue angle and the
current angular deviation from the cusp point. This process results in a
181-term vector that represents the image gamut for the current hue angle.
Appendix C – Gamut Specification Code

This IDL code is used to generate the mountain-range gamut specified in Section 2.

```idl
function make_gamut_meas_data, $
   inputGamutDataFilename, $
   N,$
   delta,$
   plot=pl,$
   smooth=sm,$
   save = sv_gmt,$
   HUE_CONVERT= hcvt
 ;+
 ; NAME:
 ; function make_gamut_meas_data, inputDataFilename, N, delta
 ;
 ; PURPOSE:
 ; This function is designed to produce a two-dimensional "mountain range",
 ; CIELAB C*, gamut from measured CIELAB values from an RGB cube. The input
 ; data needs to be ordered by FACE (i.e. According to position on the RGB
 ; cube -> Facel, Face2, ... Face6). The order needs to be maintained
 ; so that the faces can be connected using a generic triangle set for an
 ; NxM set of regular points.
 ;
 ; CATEGORY:
 ; Gamut Generation
 ;
 ; CALLING SEQUENCE:
 ;
 ; RESULT = MAKE_GAMUT_MEAS_DATA(InputDataFilename, N, Delta)
 ;
 ; INPUTS:
 ; InputDataFilename:
 ; String. Contains the name of the file that
 ; contains the raw CIELAB data for the RGB cube
 ; data for all six faces.
 ;
 ; N: The dimension of the grid used to create the RGB cube.
 ;
 ; Delta: The sampling interval used when creating the "mountain range"
 ; grid (i.e. when Delta=1.0 then the resulting grid will represent
 ; data that ranges from 0 - 360 degrees in 1 degree increments and 0
 ; - 100 L* units in 1 unit increments).
 ;
 ; OPTIONAL INPUTS:
 ; None:
 ;
 ; KEYWORD PARAMETERS:
```

C-1
; PLOT: Give a plot of the vertices and triangles used to generate the
; uniform
; grid of C* values.
;
; SMOOTH: Set this keyword to an integer value to apply a rectangular
smoothing
; function to the interpolated C* data set. The default size of the
rectangular
; smoothing function is (3). Any other value must be specified by
smooth =#in
; the calling statement.
;
; SAVE: Saves copies of gamuts for the original CIELAB points the
; made up the measured data and an interpolated gamut generated
; from the interpolation of the RGB cube grid points from the
; C* mountain range gamut.
;
; OUTPUTS:
; This function returns a two dimensional matrix of floats. The data are
; the CIELAB C* values for the surface of the gamut. The data range from
; 0 -> 360 degrees in hue angle and from 0 - 100 in normalized L* values.
;
; OPTIONAL OUTPUTS:
; None:
;
; PROCEDURE:
; T.B.D - Insert description of the algorithm here.
;
; EXAMPLE:
; CSTARGAMUT = make_gamut_meas_data('SampleCIELABDATA.dat',20,1,/,PRINT,
; SMOOTH=5)
; Description: Read the CIELAB cube data from file 'SampleCIELABDATA.dat'.
This
; file contains data from a cube with (20) nodes per side.
Sample
; the resulting grid ever (1) degree in hue angle and (1)
; lightness
; unit. Apply a 5x5 smoothing function to the results.
;
; MODIFICATION HISTORY:
; Written by: Gustav Braun, 6/20/97
; Modified: Gustav Braun, 8/19/97
; Modified to include an option to create a gamut for the visually
; corrected CIELAB hue angle. This function is activated by the
; KEYWORD selection /HUE_CONVERT.
;
; 2/15/99 Changed function to use DELAUNEY triangulation
instead of
; RGB connectivity list for interpolation. This gives a better
; LCH tesselation than the rgb connectivity.
ON_ERROR, 1 ; What should I do if an error happens? Exit to command level.

** Check the number of parameters in the argument list passed in by the user. If this number
** does not match your function prototype then return an error message and exit the function.

CASE N_PARAMS() of

3: BEGIN

    numPtsPerFace = N*N

    ; * NOTE: The data in Lab is broken up by faces every (numPtsPerFace)
    ; * Therefore the first (numPtsPerFace) are Face 1 data (i.e. 0 ->
    (numPtsPerFace-1))
    ; * To select a given face simply index by the face number and the
    (numPtsPerFace)
    ; * (i.e. the index values on the 2nd Face are (numPtsPerFace) ->
    (2*numPtsPerFace-1)

    ; * Read in the measured CIELAB data for the gamut surface
    Lab = fltarr(3,6*numPtsPerFace)
    openr, 1,inputGamutDataFilename
    readf, 1,Lab
    close, 1

    ;********** Convert CIELAB data to LCh coordinates
    LCh = lab21ch(Lab)

;******************************************************************************

;CONVERT THE CIELAB HUE INTO CORRECTED HUE SPACE
 IF keyword_set(hcvt) THEN BEGIN
   restore,'~/research/gamutFiles/CIELAB_HUE_LUTS.idl_data'
   VIS_CIELAB_Hue = interpolate(CIELAB2CIELAB_VIS,LChT2,LCh(l,*))
   LCh(2,*) = VIS_CIELAB_Hue
 ENDIF

;******************************************************************************

;* Define the min and max lightness values for the input data and use these
values to normalize the LCh values.
;* Do this so that the interpolation process only happens over the range of
data present.

    minL = min(LCh(0,*),min_Index)
    maxL = max(LCh(0,*),max_Index)

;* Force the chroma of the black point to zero
Lch(1,min_Index) = 0.0

;* Normalize the lightness channel between L*=0 and L*=100
Lch(0,*) = (Lch(0,*) - minL) / (maxL-minL) * 100.0

;* Determine if each face crosses the positive a* axis.
;* If the face crosses the +a* axis then add 360 degrees to the hue angle of
the points that are in the -a*, -b* ;* quadrant so that the projected [h,L*] data can be triangulated without
having the triangles fold back
;* on each other because of the hue angle wrap-around at the +a* axis (i.e.
0=360).

Facel = Lch(*,0:numPtsPerFace-1)

lowHueAngleIndex = where(Facel(2,*) LE 10 and Facel(2,*) GT 0 , countlow)
highHueAngleIndex = where(Facel(2,*) GE 330 and Facel(2,*) LT 360, counthigh)

;IF (lowHueAngleIndex NE (-1) and highHueAngleIndex NE (-1)) then begin
IF (countlow NE 0 AND counthigh NE 0) THEN BEGIN
indexWrap = where(Facel(2,*) LT 180)
Facel(2,indexWrap) = Facel(2,indexWrap) + 360.0
ENDIF

Face2 = Lch(*,numPtsPerFace:(2*numPtsPerFace-1))

lowHueAngleIndex = where(Face2(2,*) LE 10 and Face2(2,*) GT 0, countlow)
highHueAngleIndex = where(Face2(2,*) GE 330 and Face2(2,*) LT 360, counthigh)

;IF (lowHueAngleIndex NE (-1) and highHueAngleIndex NE (-1)) then begin
IF (countlow NE 0 AND counthigh NE 0) THEN BEGIN
indexWrap = where(Face2(2,*) LT 180)
Face2(2,indexWrap) = Face2(2,indexWrap) + 360.0
ENDIF

Face3 = Lch(*,2*numPtsPerFace:(3*numPtsPerFace-1))

lowHueAngleIndex = where(Face3(2,*) LE 10 and Face3(2,*) GT 0, countlow)
highHueAngleIndex = where(Face3(2,*) GE 330 and Face3(2,*) LT 360, counthigh)

;IF (lowHueAngleIndex NE (-1) and highHueAngleIndex NE (-1)) then begin
IF (countlow NE 0 AND counthigh NE 0) THEN BEGIN
indexWrap = where(Face3(2,*) LT 180)
Face3(2,indexWrap) = Face3(2,indexWrap) + 360.0
ENDIF
Face4 = Lch(*,3*numPtsPerFace:(4*numPtsPerFace-1))

lowHueAngleIndex = where(Face4(2,*) LE 10 and Face4(2,*) GT 0, countlow)
highHueAngleIndex = where(Face4(2,*) GE 330 and Face4(2,*) LT 360,
counthigh)

;IF (lowHueAngleIndex NE (-1) and highHueAngleIndex NE (-1)) then begin
  IF (countlow NE 0 AND counthigh NE 0) THEN BEGIN
    indexWrap = where(Face4(2,*) LT 180)
    Face4(2,indexWrap) = Face4(2,indexWrap) + 360.0
  ENDIF

Face5 = Lch(*,4*numPtsPerFace:(5*numPtsPerFace-1))

lowHueAngleIndex = where(Face5(2,*) LE 10 and Face5(2,*) GT 0, countlow)
highHueAngleIndex = where(Face5(2,*) GE 330 and Face5(2,*) LT 360,
counthigh)

;IF (lowHueAngleIndex NE (-1) and highHueAngleIndex NE (-1)) then begin
  IF (countlow NE 0 AND counthigh NE 0) THEN BEGIN
    indexWrap = where(Face5(2,*) LT 180)
    Face5(2,indexWrap) = Face5(2,indexWrap) + 360.0
  ENDIF

Face6 = Lch(*,5*numPtsPerFace:(6*numPtsPerFace-1))

lowHueAngleIndex = where(Face6(2,*) LE 10 and Face6(2,*) GT 0, countlow)
highHueAngleIndex = where(Face6(2,*) GE 330 and Face6(2,*) LT 360,
counthigh)

;IF (lowHueAngleIndex NE (-1) and highHueAngleIndex NE (-1)) then begin
  IF (countlow NE 0 AND counthigh NE 0) THEN BEGIN
    indexWrap = where(Face6(2,*) LT 180)
    Face6(2,indexWrap) = Face6(2,indexWrap) + 360.0
  ENDIF

;* Group all of the Face LCh data back into one matrix that will be used to
generate the regular grid of LCh data
LCh_faces = [Face1, Face2, Face3, Face4, Face5, Face6]

;* Make a 360 degree periodic copy of the (LCh_faces) data. This will make a
smooth transition between the
;* 0 degree and 360 degree interface.
LCh_faces2 = LCh_faces
LCh_faces2(2,*) = LCh_faces2(2,*) - 360.0

;* Make a triangle list that has the same number of points as the RGB cube
used to generate the sample data
trl = gen_cube_triangles(N,N)

s=size(Facel)

tr = [[trl],[trl + s(2)],[trl + 2*s(2)],[trl + 3*s(2)],[trl + 4*s(2)],[trl + 5*s(2)]]

LCh_periodic = [[LCh_faces],[LCh_faces2]]

triangulate, LCh_periodic(2,*), LCh_periodic(0,*), tr

s=size(tr)

if keyword_set(pl) then begin

window,5,retain=2
plot, LCh_periodic(2,*), LCh_periodic(0,0:*), psym=3, xrange=[-450,450]

for i=0L, s(2)-1 DO BEGIN

   t = [tr(*,i), tr(0,i)]
   oplot, LCh_periodic(2,t), LCh_periodic(0,t)
   oplot, LCh_Faces(2,t)-360.0, LCh_Faces(0,t)

ENDFOR

ENDIF

;* Generate a new set of points that contain both the shifted original points
; as well as a periodic
;* copy of this set that will be used to interpolate out the regular grid.
The periodic set is used
;* to maintain the continuity of at the edged of the gamut (i.e. 0 degrees
; and 360 degrees).

LCh_periodic = [[LCh_faces],[LCh_faces2]]

;********END MODIFICATION 2/15/99**************

;* Generate a new trinagle list that incorporates the periodic data as well
; as the original data. The indicies
;* of the verticies for the second half of the trinagle list need to be
; shifted by the number of data points in
;* the set.

tr_periodic = [[tr],[tr+s(2)]]

;********END MODIFICATION 2/15/99**************
Given the triangle list and the adjusted set of LCh values for the vertices of the RGB cube, interpolate
out a uniform grid of C* values over the hue angle region of 0 -> 360 degrees in increments specified
by the (delta) parameter passed into the function.

`interpVals = trigrid(LCh_periodic(2,*), LCh_periodic(0,*), $ LCh_periodic(1,*), tr_periodic, [delta, delta], [0, 0, 360, 100])`

Generate a smoothed version of the C* gamut when the /SMOOTH=sm keyword is selected

IF keyword_set(sm) THEN BEGIN
  IF sm EQ 1 THEN $ interpVals = smooth(interpVals, 3) $
  ELSE $ interpVals = smooth(interpVals, sm) $
ENDIF

WRITE A COPY OF ALL GAMUTS TO DISK

IF keyword_set(sv_gmt) THEN BEGIN
  filename = 'mountRnge_m_g_m_d_'
  f1 = make_non_lin_index_face(N)
  s = size(f1)
  zeros = f1tarr(1,s(2))
  face1 = [zeros, f1]
  face2 = [f1(0,*), zeros, f1(1,*)]
  face3 = [f1, zeros]
  face4 = [zeros + 1, f1]
  face5 = [f1(0,*), zeros+1, f1(1,*)]
  face6 = [f1, zeros+1]

  writesurfgeom, lch21ab(LCh), [[face1], [face2], [face3], [face4], [face5], [face 6]], $ long(tr), filename="_raw_data.geom"

  interpCstar = interpolate(interpVals, LCh(2,*)/delta, $ LCh(0,*)/delta, /cubic)

  IF keyword_set(pl) THEN BEGIN
    window, 1, retain=2
    plot, LCh(2,*), LCh(1,*) - interpCstar, psym=3
  ENDIF
IF keyword_set(sm) THEN $

\text{writesurfgeom, lch21ab([LCh(0,*)}, \text{interpCstar,LCh(2,*))}, [[\text{face1}, [\text{face2}, [\text{face3}, [\text{face4}, [\text{face5}, [\text{face6}]),}$

\text{long(tr),filename+"_interp_data_sm"+strtrim(string(sm),1)+"_d"+strtrim(string(delta),1)+".geom"}$

ELSE $

\text{writesurfgeom, lch21ab([LCh(0,*)}, \text{interpCstar,LCh(2,*))}, [[\text{face1}, [\text{face2}, [\text{face3}, [\text{face4}, [\text{face5}, [\text{face6}]),}$

\text{long(tr),filename+"_interp_data"+_d"+strtrim(string(delta),1)+".geom"$}

ENDIF

END ;* END statement for the CASE statement

ELSE: MESSAGE, 'Wrong Number of Arguments. Try Again or Refer to Documentation using DOC_LIBRARY'

ENDCASE

return, interpVals

END ;End the procedure
Appendix D - Hue-Correction LUTs: Blue Correction Only

In order to complete the Hung and Berns (1995) data set, so that it could be used to generate a LUT, the following modifications to their data set were made. First the data were linearly extrapolated back to the origin following a line that connected the first point in each data series to \([a^*,b^*] = [0,0]\). Secondly, each data series was linearly extrapolated out to a point that had a \(C^* = 150\). This was accomplished by generating a line equation for the last two points in the data series. The intersection points between the line and a circle with radius, \(r = 150\), were determined by solving the following linear and quadratic equations:
\( b^* = ma^* + \beta \quad \text{Eqn. (1) of line connecting last 2 points in series.} \)

\((a^*)^2 + (b^*)^2 = r^2 \quad \text{Eqn. (2) of circle with radius, } r = 150.\)

Set the Eqn. (1) and Eqn. (2) equal to each other and solve for \(a^*\) and \(b^*\).

\[(a^*)^2 + (ma^* + \beta)^2 - r^2 = 0\]

\[(a^*)^2 + m^2(a^*)^2 + 2m\beta a^* + (\beta^2 - r^2) = 0\]

\[(m^2 + 1)(a^*)^2 + 2m\beta a^* + (\beta^2 - r^2) = 0\]

\[a^* = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}\]

where:

\[a = (m^2 + 1),\]

\[b = 2m\beta,\]

\[c = (\beta^2 - r^2)\]

The following Matlab code was used to generate the hue-correction LUTs used in this thesis. The final output LUTs are floating point ASCII files that can be used to convert to-and-from the hue-corrected space.

****************************************************************************************
****
This Matlab code was written to run under Matab 4.2
Purpose:
This program uses the constant lightness data from Hung and Berns (1995) to generate two LUTs specify the transformation from CIELAB to the Hung&Berns CIELAB.

\% Inputs:
All of the data needed to perform this calculation are given in the function.

\% Output:
The output is two ascii files that contain the forward and inverse LUTs.
- Forward.lut -> transforms from CIELAB-to-Hung&Berns CIELAB
- Inverse.lut -> transforms from Hung&Berns CIELAB-to-CIELAB

****************************************************************************************
% Required Functions:  
% lab_2_lch.m  
% Written By:  Gustav Braun  
% Date:  2/25/98  
%  
%****************************************************************************************

****  
% Linear hue lines  
redXYZ = [36.03 30.90 24.48;41.95 30.90 15.28;48.55 30.90 9.19;54.45 30.90 2.54];
redyellowXYZ = [60.78 61.38 52.57; 63.03 61.38 35.73; 65.02 61.38 20.64; 68.22 61.38 7.91];
yellowXYZ = [93.75 100.41 89.27; 91.17 100.41 62.45; 88.00 100.41 38.11; 86.15 100.41 14.92];
yellowGreenXYZ = [60.78 61.38 52.57; 63.03 61.38 35.73; 65.02 61.38 20.64; 68.22 61.38 7.91];
greenXYZ = [64.22 77.65 69.76; 54.54 77.65 46.58; 45.32 77.65 27.70; 35.93 77.65 13.95];
greenCyanXYZ = [74.58 84.76 94.56; 66.21 84.76 92.61; 58.29 84.76 91.06; 50.38 84.76 86.02];
cyanXYZ = [65.51 72.27 91.62; 60.35 72.27 56.33; 55.31 72.27 108.44; 49.55 72.27 114.91];
cyanBlueXYZ = [42.39 45.26 65.59; 39.95 45.26 77.71; 37.85 45.26 91.85; 37.25 45.26 111.12];
blueXYZ = [10.77 10.77 23.59; 11.53 10.77 39.05; 14.36 10.77 59.71; 21.68 10.77 105.09];
blueMagentaXYZ = [27.15 23.30 40.33; 32.40 23.30 56.33; 37.74 23.30 77.25; 44.79 23.30 105.53];
magentaXYZ = [42.86 36.83 55.50; 50.64 36.83 67.45; 59.35 36.83 81.76; 68.73 36.83 106.85];
magentaRedXYZ = [41.74 35.53 45.24; 49.23 35.53 51.21; 56.77 35.53 53.13; 64.36 35.53 53.56];
whiteC = [98.074 100 118.23];
redLab = XYZ2Lab(redXYZ,'whiteC');
redYellowLab = XYZ2Lab(redyellowXYZ,'whiteC');
yellowLab = XYZ2Lab(yellowXYZ,'whiteC');
yellowGreenLab = XYZ2Lab(yellowGreenXYZ,'whiteC');
greenLab = XYZ2Lab(greenXYZ,'whiteC');
greenCyanLab = XYZ2Lab(greenCyanXYZ,'whiteC');
cyanLab = XYZ2Lab(cyanXYZ,'whiteC');
cyanBlueLab = XYZ2Lab(cyanBlueXYZ,'whiteC');
blueLab = XYZ2Lab(blueXYZ,'whiteC');
blueMagentaLab = XYZ2Lab(blueMagentaXYZ,'whiteC');
magentaLab = XYZ2Lab(magentaXYZ,'whiteC');
magentaRedLab = XYZ2Lab(magentaRedXYZ,'whiteC');

% Add a sample at the origin for each hue line with the same L* as the hue line
redLab = [redLab(1,1)  0  0;redLab];
redYellowLab = [redYellowLab(1,1)  0  0;redYellowLab];
yellowLab = [yellowLab(1,1)  0  0;yellowLab];
yellowGreenLab = [yellowGreenLab(1,1)  0  0;yellowGreenLab];
% Perform the extrapolation out to a C* = 150

% red
x1 = redLab(5,2);
y1 = redLab(5,3);
x2 = redLab(4,2);
y2 = redLab(4,3);
m = (y1-y2)/(x1-x2);
intercept = y1 - m * x1;

r = 150;
a = (m^2 + 1);
b = 2 * intercept * m;
c = (intercept^2 - r^2);
redX1 = (-b + sqrt(b^2 - 4*a*c)) / (2*a);
redX2 = (-b - sqrt(b^2 - 4*a*c)) / (2*a);
redY1 = m * redX1 + intercept;
redY2 = m * redX2 + intercept;
redXY1 = [redX1, redY1]
redLab(6,:) = [redLab(1,1), redXY1];
refLCh = lab_2_lch(redLab(2,:)')';
refLCh(2) = 150;
refLab = lch_2_lab(refLCh')';

% redYellow
x1 = redYellowLab(5,2);
y1 = redYellowLab(5,3);
x2 = redYellowLab(4,2);
y2 = redYellowLab(4,3);
m = (y1-y2)/(x1-x2);
intercept = y1 - m * x1;

r = 150;
a = (m^2 + 1);
b = 2 * intercept * m;
c = (intercept^2 - r^2);
redYellowX1 = (-b + sqrt(b^2 - 4*a*c)) / (2*a);
redYellowX2 = (-b - sqrt(b^2 - 4*a*c)) / (2*a);
redYellowY1 = m * redYellowX1 + intercept;
redYellowY2 = m * redYellowX2 + intercept;
redYellowXY1 = [redYellowX1, redYellowY1]
redYellowLab(6,:) = [redYellowLab(1,1), redYellowXY1];
refLCh = lab_2_lch(redYellowLab(2,:)')';
refLCh(2) = 150;
refLab = lch_2_lab(refLCh')';

% yellow
x1 = yellowLab(5,2);
\[
y_1 = \text{yellowLab}(5,3);
x_2 = \text{yellowLab}(4,2);
y_2 = \text{yellowLab}(4,3);
m = (y_1 - y_2)/(x_1 - x_2);
\text{intercept} = y_1 - m \times x_1;
\]
\[
r = 150;
\]
\[
a = (m^2 + 1);
b = 2 \times \text{intercept} \times m;
c = (\text{intercept}^2 - r^2);
\]
\[
yellowX_1 = (-b + \sqrt{b^2 - 4 \times a \times c}) / (2 \times a);
yellowX_2 = (-b - \sqrt{b^2 - 4 \times a \times c}) / (2 \times a);
\]
\[
yellowY_1 = m \times yellowX_1 + \text{intercept};
yellowY_2 = m \times yellowX_2 + \text{intercept};
yellowXY_1 = [yellowX_1, yellowY_1]
yellowXY_2 = [yellowX_2, yellowY_2]
yellowLab(6,:) = [yellowLab(1,1), yellowXY_2]
\]
\[
\text{refLCh} = \text{lab}_2\_\text{lch}\left(\text{yellowLab}(2,:)\right)';
\text{refLCh}(2) = 150;
\text{refLab} = \text{lch}_2\_\text{lab}(\text{refLCh}')';
\]
\[
% \text{yellowGreen}
\]
\[
x_1 = \text{yellowGreenLab}(5,2);
y_1 = \text{yellowGreenLab}(5,3);
x_2 = \text{yellowGreenLab}(4,2);
y_2 = \text{yellowGreenLab}(4,3);
m = (y_1 - y_2)/(x_1 - x_2);
\text{intercept} = y_1 - m \times x_1;
\]
\[
r = 150;
\]
\[
a = (m^2 + 1);
b = 2 \times \text{intercept} \times m;
c = (\text{intercept}^2 - r^2);
\]
\[
yellowGreenX_1 = (-b + \sqrt{b^2 - 4 \times a \times c}) / (2 \times a);
yellowGreenX_2 = (-b - \sqrt{b^2 - 4 \times a \times c}) / (2 \times a);
\]
\[
yellowGreenY_1 = m \times yellowGreenX_1 + \text{intercept};
yellowGreenY_2 = m \times yellowGreenX_2 + \text{intercept};
yellowGreenXY_1 = [yellowGreenX_1, yellowGreenY_1]
yellowGreenXY_2 = [yellowGreenX_2, yellowGreenY_2]
yellowGreenLab(6,:) = [yellowLab(1,1), yellowGreenXY_2]
\]
\[
\text{refLCh} = \text{lab}_2\_\text{lch}\left(\text{yellowGreenLab}(2,:)\right)';
\text{refLCh}(2) = 150;
\text{refLab} = \text{lch}_2\_\text{lab}(\text{refLCh}')';
\]
\[
% \text{green}
\]
\[
x_1 = \text{greenLab}(5,2);
y_1 = \text{greenLab}(5,3);
x_2 = \text{greenLab}(4,2);
y_2 = \text{greenLab}(4,3);
m = (y_1 - y_2)/(x_1 - x_2);
\text{intercept} = y_1 - m \times x_1;
\]
\[
r = 150;
\]
\[
a = (m^2 + 1);
b = 2 \times \text{intercept} \times m;
}\]
c = (intercept^2 - r^2);

greenX1 = (-b + sqrt(b^2 - 4*a*c)) / (2*a);
greenX2 = (-b - sqrt(b^2 - 4*a*c)) / (2*a);

greenY1 = m*greenX1 + intercept;
greenY2 = m*greenX2 + intercept;

greenXY1 = [greenX1, greenY1]
greenXY2 = [greenX2, greenY2]
greenLab(6,i) = [greenLab(1,1), greenXY2];

testlab = lab_2_lch( greenLab(2, :) );
testlab(2) = 150;
testlab = lch_2_lab(testlab);
refLCh(2) = 150;
refLab = lch_2_lab(refLCh');

%cyanBlue
x1 = cyanBlueLab(5,2);
y1 = cyanBlueLab(5,3);
x2 = cyanBlueLab(4,2);
y2 = cyanBlueLab(4,3);
m = (y1-y2)/(x1-x2);
intercept = y1 - m*x1;

r = 150;

a = (m^2 + 1);
b = 2*intercept*m;
c = (intercept^2 - r^2);

cyanBlueX1 = (-b + sqrt(b^2 - 4*a*c)) / (2*a);
cyanBlueX2 = (-b - sqrt(b^2 - 4*a*c)) / (2*a);

cyanBlueY1 = m*cyanBlueX1 + intercept;
cyanBlueY2 = m*cyanBlueX2 + intercept;

cyanBlueXY1 = [cyanBlueX1, cyanBlueY1];
cyanBlueXY2 = [cyanBlueX2, cyanBlueY2];
cyanBlueLab(6,:) = [cyanBlueLab(1,1), cyanBlueXY2];

refLCh = lab_2_lch(cyanBlueLab(2,:))';
refLCh(2) = 150;
refLab = lch_2_lab(refLCh');

%blue
x1 = blueLab(5,2);
y1 = blueLab(5,3);
x2 = blueLab(4,2);
y2 = blueLab(4,3);
m = (y1-y2)/(x1-x2);
intercept = y1 - m*x1;

r = 150;

a = (m^2 + 1);
b = 2*intercept*m;
c = (intercept^2 - r^2);

blueX1 = (-b + sqrt(b^2 - 4*a*c)) / (2*a);
blueX2 = (-b - sqrt(b^2 - 4*a*c)) / (2*a);

blueY1 = m*blueX1 + intercept;
blueY2 = m*blueX2 + intercept;

blueXY1 = [blueX1, blueY1];
blueXY2 = [blueX2, blueY2];
blueLab(6,:) = [blueLab(1,1), blueXY1];

refLCh = lab_2_lch(blueLab(2,:))';
refLCh(2) = 150;
refLab = lch_2_lab(refLCh');

%blueMagenta
x1 = blueMagentaLab(5,2);
y1 = blueMagentaLab(5,3);
x2 = blueMagentaLab(4,2);
y2 = blueMagentaLab(4,3);
m = (y1-y2)/(x1-x2);

intercept = y1 - m*x1;
\[ r = 150; \]
\[ a = (m^2 + 1); \]
\[ b = 2*intercept*m; \]
\[ c = (intercept^2 - r^2); \]
\[ blueMagentaX1 = (-b + \sqrt{b^2 - 4*a*c}) / (2*a); \]
\[ blueMagentaX2 = (-b - \sqrt{b^2 - 4*a*c}) / (2*a); \]
\[ blueMagentaY1 = m*blueMagentaX1 + intercept; \]
\[ blueMagentaY2 = m*blueMagentaX2 + intercept; \]
\[ blueMagentaXY1 = [blueMagentaX1, blueMagentaY1] \]
\[ blueMagentaXY2 = [blueMagentaX2, blueMagentaY2] \]
\[ blueMagentaLab(6,:) = [blueMagentaLab(1,1), blueMagentaXY1]; \]
\[ refLCh = lab_2_lch(blueMagentaLab(2,:)'); \]
\[ refLCh(2) = 150; \]
\[ refLab = lch_2_lab(refLCh'); \]
\[ %magenta \]
\[ xl = magentaLab(5,2); \]
\[ yl = magentaLab(5,3); \]
\[ x2 = magentaLab(4,2); \]
\[ y2 = magentaLab(4,3); \]
\[ m = (yl-y2)/(xl-x2); \]
\[ intercept = yl - m * xl; \]
\[ r = 150; \]
\[ a = (m^2 + 1); \]
\[ b = 2*intercept*m; \]
\[ c = (intercept^2 - r^2); \]
\[ magentaX1 = (-b + \sqrt{b^2 - 4*a*c}) / (2*a); \]
\[ magentaX2 = (-b - \sqrt{b^2 - 4*a*c}) / (2*a); \]
\[ magentaY1 = m*magentaX1 + intercept; \]
\[ magentaY2 = m*magentaX2 + intercept; \]
\[ magentaXY1 = [magentaX1, magentaY1] \]
\[ magentaXY2 = [magentaX2, magentaY2] \]
\[ magentaLab(6,:) = [magentaLab(1,1), magentaXY1]; \]
\[ refLCh = lab_2_lch(magentaLab(2,:)'); \]
\[ refLCh(2) = 150; \]
\[ refLab = lch_2_lab(refLCh'); \]
\[ %magentaRed \]
\[ xl = magentaRedLab(5,2); \]
\[ yl = magentaRedLab(5,3); \]
\[ x2 = magentaRedLab(4,2); \]
\[ y2 = magentaRedLab(4,3); \]
\[ m = (yl-y2)/(xl-x2); \]
\[ intercept = yl - m * xl; \]
\[ r = 150; \]
\[ a = (m^2 + 1); \]
\[ b = 2*intercept*m; \]
\[ c = (intercept^2 - r^2); \]
\[ magentaRedX1 = (-b + \sqrt{b^2 - 4*a*c}) / (2*a); \]
\[ magentaRedX2 = (-b - \sqrt{b^2 - 4*a*c}) / (2*a); \]
\[ magentaRedY1 = m*magentaRedX1 + intercept; \]
\[ magentaRedY2 = m*magentaRedX2 + intercept; \]
magentaRedXYl=[magentaRedXYl,magentaRedXYl]
magentaRedXY2=[magentaRedXY2,magentaRedXY2]
magentaRedLab(6,:)= [magentaRedLab(1,1),magentaRedXYl];

refLCh = lab_2_lch(yellowGreenLCh(2,:)');
refLCh(2) = 150;
refLab = lch_2_lab(refLCh');

%LCh Specify the LCh values for the hue lines. For the ones outside the
%region of color space set the hue angle as a function of chroma to the same
%Hue angle as the base hue. This will insure that the only hue correction that
%happens is in the blue region of color space.

%Hue angle constant for the all of these lines.
redLCh = [62.4231 0 40.0845;62.4231 50 40.0845];
redLCh = [62.4231 25 40.0845;62.4231 150 40.0845];
redYellowLCh=[82.5826 0 85.4921;82.5826 25 85.4921;82.5826 50 85.4921; ... 
82.5826 75 85.4921;82.5826 100 85.4921;82.5826 150 85.4921];
yellowLCh =[100.1583 0 114.1546;100.1583 25 114.1546; ... 
100.1583 75 114.1546;100.1583 100 114.1546; 
100.1583 150 114.1546];
yellowGreenLCh = [97.9214 0 133.5637;97.9214 25 133.5637; ... 
97.9214 75 133.5637;97.9214 100 133.5637;97.9214 150 133.5637];
greenLCh = [90.6200 0 147.6462;90.6200 25 147.6462;90.6200 50 147.6462; ... 
90.6200 75 147.6462;90.6200 100 147.6462;90.6200 150 147.6462];
greenCyanLCh = [93.7796 0 167.8221;93.7796 25 167.8221;93.7796 50 167.8221; ... 
93.7796 75 167.8221;93.7796 100 167.8221;93.7796 150 167.8221];
cyanLCh = [88.0984 0 199.9648;88.0984 25 199.9648;88.0984 50 199.9648; ... 
88.0984 75 199.9648;88.0984 100 199.9648];

%Hue angle varies as a function of chroma for these lines.
cyanBlueLCh = lab_2_lch(cyanBlueLab');
blueLCh = lab_2_lch(blueLab');
blueMagentaLCh = lab_2_lch(blueMagentaLab');

%Hue angle is constant as a function of chroma for these lines.
magentaLCh= [67.1493 0 330.1381;67.1493 75 330.1381;67.1493 150 330.1381];
magentaRedLCh = [66.1592 0 350.8332;66.1592 75 350.8332;66.1592 150 350.8332];

/* Generate a plot of the 12-gridlines.

figure
plot([redLCh(2,3);redLCh(2:6,3)],[0;redLCh(2:6,2)],'+r')
axis([0 360 0 150])
axis(axis)
hold on
plot([redLCh(2,3);redLCh(2:6,3)],[0;redLCh(2:6,2)],'r')
redLCh(1,2) = 0;
redLCh(1,3) = redLCh(2,3);

plot([redYellowLCh(2,3);redYellowLCh(2:6,3)],[0;redYellowLCh(2:6,2)],'y')
plot([redYellowLCh(2,3);redYellowLCh(2:6,3)],[0;redYellowLCh(2:6,2)],'y')
redYellowLCh(1,3) = redYellowLCh(2,3);

plot([yellowLCh(2,3);yellowLCh(2:6,3)],[0;yellowLCh(2:6,2)],'g')
plot([yellowLCh(2,3);yellowLCh(2:6,3)],[0;yellowLCh(2:6,2)],'g')
yellowLCh(1,2) = 0;
yellowLCh(1,3) = yellowLCh(2,3);

plot([yellowGreenLCh(2,3);yellowGreenLCh(2:6,3)],[0;yellowGreenLCh(2:6,2)],'+g')
plot([yellowGreenLCh(2,3);yellowGreenLCh(2:6,3)],[0;yellowGreenLCh(2:6,2)],'+g')
yellowGreenLCh(1,2) = 0;
yellowGreenLCh(1,3) = yellowGreenLCh(2,3);
plot([greenLCh(2,3);greenLCh(2:6,3)],[0;greenLCh(2:6,2)],'g')
greenLCh(1,2) = 0;
greenLCh(1,3) = greenLCh(2,3);
plot([greenCyanLCh(2,3);greenCyanLCh(2:6,3)],[0;greenCyanLCh(2:6,2)],'c')
greenCyanLCh(1,2) = 0;
greenCyanLCh(1,3) = greenCyanLCh(2,3);
plot([cyanLCh(2,3);cyanLCh(2:6,3)],[0;cyanLCh(2:6,2)],'c')
cyanLCh(1,2) = 0;
cyanLCh(1,3) = cyanLCh(2,3);
plot([cyanBlueLCh(2,3);cyanBlueLCh(2:6,3)],[0;cyanBlueLCh(2:6,2)],'b')
cyanBlueLCh(1,2) = 0;
cyanBlueLCh(1,3) = cyanBlueLCh(2,3);
plot([blueLCh(2,3);blueLCh(2:6,3)],[0;blueLCh(2:6,2)],'b')
blueLCh(1,2) = 0;
blueLCh(1,3) = blueLCh(2,3);
plot([blueMagentaLCh(2,3);blueMagentaLCh(2:6,3)],[0;blueMagentaLCh(2:6,2)],'m')
blueMagentaLCh(1,2) = 0;
blueMagentaLCh(1,3) = blueMagentaLCh(2,3);
plot([magentaLCh(2,3);magentaLCh(2:6,3)],[0;magentaLCh(2:6,2)],'m')
magentaLCh(1,2) = 0;
magentaLCh(1,3) = magentaLCh(2,3);
plot([magentaRedLCh(2,3);magentaRedLCh(2:6,3)],[0;magentaRedLCh(2:6,2)],'r')
magentaRedLCh(1,2) = 0;
magentaRedLCh(1,3) = magentaRedLCh(2,3);
hold off
title('Constant Visual Hue in CIELAB Blue Correction Only')
xlabel('CIELAB Hue Angle')
ylabel('C*ab')

%CALCULATE THE GRID LINES FOR THE LUT.

%red
redLChAim = redLCh;
redLChAim(:,3) = ones(6,1).*redLCh(1,3);
redDeltaHue = redLCh(:,3) - redLChAim(:,3);
redLChAim(6,2) = 150.0;
redLUT = interp1(redLChAim(:,2),redDeltaHue,0:150);
redHue = redLChAim(1,3);
disp('Hue Angle For red grid line = '),redHue

%redYellow
redYellowLChAim = redYellowLCh;
redYellowLChAim(:,3) = ones(6,1).*redYellowLCh(1,3);
redYellowDeltaHue = redYellowLCh(:,3) - redYellowLChAim(:,3);
redYellowLUT = interp1(redYellowLChAim(:,2),redYellowDeltaHue,0:150);
redYellowLUT = redYellowLChAim(1,3) + redYellowLUT;
redYellowHue = redYellowLChAim(1,3);
disp('Hue Angle For redYellow grid line = '),redYellowHue

%yellow
yellowLChAim = yellowLCh;
yellowLChAim(:,3) = ones(6,1).*yellowLCh(:,3);
yellowDeltaHue = yellowLCh(:,3) - yellowLChAim(:,3);
yellowLUT = interp1(yellowLChAim(:,2),yellowDeltaHue,0:150);
yellowLUT = yellowLChAim(1,3) + yellowLUT;
yellowHue = yellowLChAim(1,3);
disp('Hue Angle For yellow grid line = '),yellowHue

%yellowGreen
yellowGreenLChAim = yellowGreenLCh;
yellowGreenLChAim(:,3) = ones(6,1).*yellowGreenLCh(:,3);
yellowGreenDeltaHue = yellowGreenLCh(:,3) - yellowGreenLChAim(:,3);
yellowGreenLUT = interp1(yellowGreenLChAim(:,2),yellowGreenDeltaHue,0:150);
yellowGreenLUT = yellowGreenLChAim(1,3) + yellowGreenLUT;
yellowGreenHue = yellowGreenLChAim(1,3);
disp('Hue Angle For yellowGreen grid line = '),yellowGreenHue

%green
greenLChAim = greenLCh;
greenLChAim(:,3) = ones(6,1).*greenLCh(:,3);
greenDeltaHue = greenLCh(:,3) - greenLChAim(:,3);
greenLUT = interp1(greenLChAim(:,2),greenDeltaHue,0:150);
greenLUT = greenLChAim(1,3) + greenLUT;
greenHue = greenLChAim(1,3);
disp('Hue Angle For green grid line = '),greenHue

%cyan
cyanLChAim = cyanLCh;
cyanLChAim(:,3) = ones(6,1).*cyanLCh(:,3);
cyanDeltaHue = cyanLCh(:,3) - cyanLChAim(:,3);
cyanLUT = interp1(cyanLChAim(:,2),cyanDeltaHue,0:150);
cyanLUT = cyanLChAim(1,3) + cyanLUT;
cyanHue = cyanLChAim(1,3);
disp('Hue Angle For cyan grid line = '),cyanHue

%blue
cyanBlueLChAim = cyanBlueLCh;
cyanBlueLChAim(:,3) = ones(6,1).*cyanBlueLCh(:,3);
cyanBlueDeltaHue = cyanBlueLCh(:,3) - cyanBlueLChAim(:,3);
cyanBlueLUT = interp1(cyanBlueLChAim(:,2),cyanBlueDeltaHue,0:150);
cyanBlueLUT = cyanBlueLChAim(1,3) + cyanBlueLUT;
cyanBlueHue = cyanBlueLChAim(1,3);
disp('Hue Angle For cyanBlue grid line = '),cyanBlueHue
%blueMagenta

blueMagentaLChAim = blueMagentaLCh;
blueMagentaLChAim(:,3) = ones(61).*blueMagentaLCh(1,3);
blueMagentaDeltaHue = blueMagentaLCh(:,3) - blueMagentaLChAim(:,3);
blueMagentaLUT = interp1(blueMagentaLChAim(:,2),blueMagentaDeltaHue,0:150);
blueMagentaLUT = blueMagentaLChAim(1,3) + blueMagentaLUT;
blueMagentaHue = blueMagentaLChAim(1,3);
disp('Hue Angle For blueMagenta grid line = '),blueMagentaHue

%magenta

magentaLChAim = magentaLCh;
magentaLChAim(:,3) = ones(61).*magentaLCh(1,3);
magentaDeltaHue = magentaLCh(:,3) - magentaLChAim(:,3);
magentaLUT = interp1(magentaLChAim(:,2),magentaDeltaHue,0:150);
magentaLUT = magentaLChAim(1,3) + magentaLUT;
magentaHue = magentaLChAim(1,3);
disp('Hue Angle For magenta grid line = '),magentaHue

%magentaRed

magentaRedLChAim = magentaRedLCh;
magentaRedLChAim(:,3) = ones(61).*magentaRedLCh(1,3);
magentaRedDeltaHue = magentaRedLCh(:,3) - magentaRedLChAim(:,3);
magentaRedLUT = interp1(magentaRedLChAim(:,2),magentaRedDeltaHue,0:150);
magentaRedLUT = magentaRedLChAim(1,3) + magentaRedLUT;
magentaRedHue = magentaRedLChAim(1,3);
disp('Hue Angle For magentaRed grid line = '),magentaRedHue

%FILL THE 2D LUT GRID USING THE GRID LINES AND LINEAR INTERPOLATION BETWEEN THE GRIDLINES
%FOR EACH CHROMA VALUE

LUT = zeros(151,361); %LUT that goes from hue-corrected space to CIELAB
LUT Forward = zeros(151,361); %LUT that goes from CIELAB to hue-corrected space.
temp = [redHue;redYellowHue;yellowHue;yellowGreenHue;greenHue;greenCyanHue; ...
cyanHue;cyanBlueHue;
blueHue;blueMagentaHue;magentaRedHue];
gridLinesHue = temp 360; temp + 360; %Makes the data periodic so that 360-degrees matches 0-degrees

for i = 1:151

temp = [redLUT(i);redYellowLUT(i);yellowLUT(i);yellowGreenLUT(i);greenLUT(i); ...
greenCyanLUT(i);cyanLUT(i);
cyanBlueLUT(i);blueLUT(i);blueMagentaLUT(i); ...
magentaLUT(i);magentaRedLUT(i)];

tempGridLines = [temp 360;temp+360];
LUT(:,i) = interp1(gridLinesHue,tempGridLines,0:360);
LUT Forward(:,i) = interp1(tempGridLines,gridLinesHue,0:360);"
end

% Generate a plot of the resulting forward and inverse LUT's every 8-degrees
figure
hold on
plot(LUT(:,1:8:361),[0:150]'*ones(1,46), 'w')
xlabel('CIELAB Hue Angle')
ylabel('C*ab')
title('Inverse LUT: Hung & Berns - CIELAB-to-CIELAB')
hold off

figure
plot(LUT_Forward(:,1:8:361),[0:150]'*ones(1,46), 'w')

title('Forward LUT CIELAB-to-Hung & Berns-CIELAB')
xlabel('Visual CIELAB HUE Angle')
ylabel('C*ab')

% Save the LUTs as ASCII files.
save 'Forward.lut' LUT_Forward -ascii -tabs
save 'Inverse.lut' LUT -ascii -tabs
Appendix E - Hung and Berns (1995) Constant-Lightness data

Table E.1 Listing of the hue angle and chroma data from the Hung and Berns (1995) constant lightness data set. Included are the base hue angles for the 12 gridlines used in the hue-correction transformations.

<table>
<thead>
<tr>
<th>Color name</th>
<th>C'uv</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>36.03</td>
<td>30.90</td>
<td>24.48</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>41.95</td>
<td>30.90</td>
<td>15.29</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>48.55</td>
<td>30.90</td>
<td>9.19</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>54.45</td>
<td>30.90</td>
<td>7.54</td>
<td></td>
</tr>
<tr>
<td>Red-yellow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>60.78</td>
<td>61.38</td>
<td>52.57</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>63.03</td>
<td>61.38</td>
<td>35.73</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>65.02</td>
<td>61.38</td>
<td>20.64</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>68.22</td>
<td>61.38</td>
<td>7.91</td>
<td></td>
</tr>
<tr>
<td>Yellow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>93.75</td>
<td>100.41</td>
<td>89.27</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>91.17</td>
<td>100.41</td>
<td>62.45</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>88.00</td>
<td>100.41</td>
<td>38.11</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>86.15</td>
<td>100.41</td>
<td>14.92</td>
<td></td>
</tr>
<tr>
<td>Yellow-green</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>84.07</td>
<td>94.72</td>
<td>59.31</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>77.72</td>
<td>94.72</td>
<td>36.22</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>71.61</td>
<td>94.72</td>
<td>15.28</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>65.02</td>
<td>94.72</td>
<td>7.91</td>
<td></td>
</tr>
<tr>
<td>Green</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>64.22</td>
<td>77.65</td>
<td>67.36</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>54.64</td>
<td>77.65</td>
<td>35.73</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>45.32</td>
<td>77.65</td>
<td>27.70</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>35.93</td>
<td>77.65</td>
<td>13.93</td>
<td></td>
</tr>
<tr>
<td>Green-cyan</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>74.58</td>
<td>94.72</td>
<td>94.56</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>60.21</td>
<td>84.76</td>
<td>92.61</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>58.29</td>
<td>84.76</td>
<td>91.06</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>50.38</td>
<td>84.76</td>
<td>86.02</td>
<td></td>
</tr>
<tr>
<td>Cyan</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>65.51</td>
<td>72.72</td>
<td>91.62</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>60.35</td>
<td>72.72</td>
<td>99.51</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>55.31</td>
<td>72.72</td>
<td>108.44</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>49.55</td>
<td>72.72</td>
<td>114.91</td>
<td></td>
</tr>
<tr>
<td>Cyan-blue</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>42.59</td>
<td>45.26</td>
<td>85.50</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>39.95</td>
<td>45.26</td>
<td>77.71</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>37.85</td>
<td>45.26</td>
<td>91.85</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>37.25</td>
<td>45.26</td>
<td>111.12</td>
<td></td>
</tr>
<tr>
<td>Blue</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>10.77</td>
<td>10.77</td>
<td>23.69</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>11.53</td>
<td>10.77</td>
<td>59.71</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>14.35</td>
<td>10.77</td>
<td>108.44</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>21.68</td>
<td>10.77</td>
<td>114.91</td>
<td></td>
</tr>
<tr>
<td>Blue-magenta</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>27.15</td>
<td>23.30</td>
<td>40.33</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>32.40</td>
<td>23.30</td>
<td>56.33</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>37.74</td>
<td>23.30</td>
<td>77.25</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>44.79</td>
<td>23.30</td>
<td>105.63</td>
<td></td>
</tr>
<tr>
<td>Magenta</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>42.86</td>
<td>36.83</td>
<td>55.50</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>50.64</td>
<td>36.83</td>
<td>57.54</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>59.35</td>
<td>36.83</td>
<td>81.76</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>68.73</td>
<td>36.83</td>
<td>106.85</td>
<td></td>
</tr>
<tr>
<td>Magenta-red</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>41.74</td>
<td>35.53</td>
<td>45.24</td>
<td></td>
</tr>
<tr>
<td>2/4</td>
<td>49.23</td>
<td>35.53</td>
<td>51.21</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>56.77</td>
<td>35.53</td>
<td>53.13</td>
<td></td>
</tr>
<tr>
<td>Ref.</td>
<td>64.39</td>
<td>35.53</td>
<td>53.66</td>
<td></td>
</tr>
</tbody>
</table>
Appendix F – Hue-Correction Experiment Images

The image shown in Figure F-1 – F-5 were used in the hue-correction experiments given in Section 3.

Figure F-1. Amsterdam image. Contained a mixture of hues that required gamut mapping.

Figure F-2. Capital image. Contained predominantly blue hues.
Figure F-3. Pool-ball image. Contained a mixture of hues that were gamut mapped.  
(Note: Blue was the most affected by the gamut mapping.)

Figure F-4. Mushroom image. Contained predominantly red hues.
Figure F-5. Macaws image. Contained predominantly red hues that were gamut mapped.
Appendix G –Hue-Correction Experiment Interval Scales

The interval scales shown in Figure G-1 – G-5 are for each image and algorithm tested in the hue-correction experiments performed in Section 3. The three color spaces tested were CIELAB, the Hung and Berns hue-corrected CIELAB (H&B CIELAB) and the Ebner and Fairchild color space given in Section 3. The three gamut-mapping algorithms tested were constant-lightness chroma clipping (CCLP), minimum ΔE_{ab*} clipping (MnDE), and centroid clipping (CP2P). (Note: All gamut-mapping algorithms preserved metric hue angle in the reference color space.)

*Figure G-1. Interval scales for the Macaws image from the hue-correction experiment performed in Section 3.*
Figure G-2. Interval scales for the Mushroom image from the hue-correction experiment performed in Section 3.

Figure G-3. Interval scales for the Amsterdam image from the hue-correction experiment performed in Section 3.
Figure G-4. Interval scales for the Capital image from the hue-correction experiment performed in Section 3.

Figure G-5. Interval scales for the Pool-ball image from the hue-correction experiment performed in Section 3.
Appendix H – Hue-Correction Look-up Tables

The look-up tables (LUTs) that were used to transform from CIELAB to the Hung and Berns hue-corrected CIELAB color space and from the hue-corrected space back to CIELAB have been posted on the Munsell Color Science Laboratory (MCSL) web page. They can be accessed from the main MCSL web page at http://www.cis.rit.edu/mcsl/

From this level follow the links for the student research pages.

The LUTs are stored in plain text, tab delimited ASCII text files. The file forwardLUT.txt contains the LUT that transforms from CIELAB hue to the Hung and Berns corrected hue. The file inverseLUT.txt contains the LUT that transforms from Hung and Berns corrected hue to CIELAB hue. The data from these tables can be used to estimate the destination hue for any given input color, specified by its \([C_{ab}^*, h_{ab}]\) coordinates, utilizing bilinear interpolation.
Appendix I – Observer Results from Sigmoidal Screening Experiment

Table I.1. The values in this table represent the number of times a given remapped image was selected as the closest match to the original for the Phase 2 screening process. The highlighted cells indicate the top three sigmoidal remapping functions ($S_1$-$S_3$) that were selected the most often. (If two images had the same tally they were both highlighted.) The images corresponding to the highlighted cells were tested in Phase 3.

<table>
<thead>
<tr>
<th>Image Name</th>
<th>$L^*$ <em>minOut</em></th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
<th>$S_4$</th>
<th>$S_5$</th>
<th>$S_6$</th>
<th>$S_7$</th>
<th>$S_8$</th>
<th>$S_9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Couple-On-Beach</td>
<td>10</td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>8</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Couple-On-Beach</td>
<td>15</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>3</td>
<td>8</td>
<td>0</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Couple-On-Beach</td>
<td>20</td>
<td>6</td>
<td>2</td>
<td>7</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Couple-On-Beach</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>8</td>
<td>3</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Flower</td>
<td>10</td>
<td>5</td>
<td>2</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Flower</td>
<td>15</td>
<td>6</td>
<td>2</td>
<td>8</td>
<td>6</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Flower</td>
<td>20</td>
<td>7</td>
<td>4</td>
<td>8</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Flower</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>7</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Horse-Race</td>
<td>10</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Horse-Race</td>
<td>15</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Horse-Race</td>
<td>20</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Horse-Race</td>
<td>5</td>
<td>8</td>
<td>5</td>
<td>7</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Macaws</td>
<td>10</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Macaws</td>
<td>15</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>7</td>
<td>2</td>
<td>7</td>
<td>2</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Macaws</td>
<td>20</td>
<td>7</td>
<td>4</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Macaws</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>7</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Raft</td>
<td>10</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Raft</td>
<td>15</td>
<td>7</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Raft</td>
<td>20</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Raft</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Temple</td>
<td>10</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Temple</td>
<td>15</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Temple</td>
<td>20</td>
<td>6</td>
<td>4</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Temple</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>3</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>
Appendix J – Sigmoidal Lightness-Remapping Experiment

Interval Scales (Phase 3)

Table J.1. Interval scales from the Phase 3 visual experiment. For each image at each of the four $L_{\text{min}}^*$ levels, the highlighted cell indicates which of the $(S_1-S_9)$ curves produced the best match to the original. Empty cells indicate that the image was not involved in this test.

<table>
<thead>
<tr>
<th>Image Name</th>
<th>$L_{\text{min}}^*$</th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
<th>$S_4$</th>
<th>$S_5$</th>
<th>$S_6$</th>
<th>$S_7$</th>
<th>$S_8$</th>
<th>$S_9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Couple-On-</td>
<td>10</td>
<td>-0.253</td>
<td>0.456</td>
<td>-0.842</td>
<td>0.639</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Beach</td>
<td>15</td>
<td>-0.399</td>
<td>0.379</td>
<td>-0.379</td>
<td>-0.637</td>
<td>-</td>
<td></td>
<td></td>
<td>1.035</td>
<td></td>
</tr>
<tr>
<td>Couple-On-</td>
<td>20</td>
<td>-0.132</td>
<td>-0.121</td>
<td>0.379</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Beach</td>
<td>5</td>
<td></td>
<td>0.507</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-0.126</td>
</tr>
<tr>
<td>Flower</td>
<td>10</td>
<td>-0.652</td>
<td></td>
<td>1.284</td>
<td>-0.640</td>
<td>0.001</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flower</td>
<td>15</td>
<td>-0.379</td>
<td></td>
<td>0.637</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flower</td>
<td>20</td>
<td>0.397</td>
<td></td>
<td></td>
<td>0.399</td>
<td>0.639</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flower</td>
<td>5</td>
<td>0.289</td>
<td></td>
<td></td>
<td></td>
<td>0.934</td>
<td>-0.948</td>
<td>-0.024</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horse-Race</td>
<td>10</td>
<td>-0.128</td>
<td></td>
<td>0.379</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-0.251</td>
</tr>
<tr>
<td>Horse-Race</td>
<td>15</td>
<td>-0.260</td>
<td>0.000</td>
<td>0.260</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horse-Race</td>
<td>20</td>
<td>-0.126</td>
<td></td>
<td>0.650</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-0.524</td>
</tr>
<tr>
<td>Horse-Race</td>
<td>5</td>
<td>-0.132</td>
<td>-0.507</td>
<td>0.639</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macaws</td>
<td>10</td>
<td>-0.006</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macaws</td>
<td>15</td>
<td>-0.251</td>
<td>1.179</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macaws</td>
<td>20</td>
<td>0.002</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macaws</td>
<td>5</td>
<td>0.385</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Raft</td>
<td>10</td>
<td>0.399</td>
<td>0.126</td>
<td>-0.260</td>
<td>-0.139</td>
<td>-0.126</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Raft</td>
<td>15</td>
<td>0.145</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Raft</td>
<td>20</td>
<td>-0.859</td>
<td>1.767</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Raft</td>
<td>5</td>
<td>-0.253</td>
<td>-0.639</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Temple</td>
<td>10</td>
<td>-0.128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Temple</td>
<td>15</td>
<td>-0.271</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Temple</td>
<td>20</td>
<td>-0.013</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Temple</td>
<td>5</td>
<td>-0.132</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix K – Sigmoidal Lightness-Remapping Model

Parameters

Table K.1. Optimal $x_o$ and $\Sigma$ parameters for the three image lightness classes and $L^*_{minOut}$ levels from Phase 3.

<table>
<thead>
<tr>
<th>High Lightness-class</th>
<th>Normal Lightness-class</th>
<th>Low Lightness-class</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L^*_{minOut}$</td>
<td>$x_o$</td>
<td>$\Sigma$</td>
</tr>
<tr>
<td>5</td>
<td>54.0</td>
<td>44.1</td>
</tr>
<tr>
<td>10</td>
<td>61.7</td>
<td>46.4</td>
</tr>
<tr>
<td>15</td>
<td>68.0</td>
<td>47.5</td>
</tr>
<tr>
<td>20</td>
<td>71.9</td>
<td>47.5</td>
</tr>
</tbody>
</table>
Appendix L – Monitor-to-Print Experiment Images

The images shown in Figure L-1 – L-4 were used in the gamut-mapping experiment performed in Section 5.

Figure L-1a,b. Boy-blue-sweater and Family-by-car images.

Figure L-2a,b. Flying-bride and Kid-in-tire images.
Figure L-3a,b. Ladies-yellow-wall and Wine-man images.

Figure L-4. Logger image.
Appendix M - Scanner Characterization and Error Analysis

The model developed for the scanner characterization used in the print-to-print experiments performed in Section 5 were adopted from the process given by Berns (1994). The only difference between the model described below and that given by Berns (1994) was that the gray-balancing process was performed with respect to CIELAB L* rather than luminance. (Note: Performing the gray-balancing in this manner gave slightly better results.)

Model Development

1. Convert measured reflectance factors to XYZ using ASTM D50 spectral weights.

2. Normalize the XYZs to the input maximum and minimum.

3. Convert to CIELAB.

4. Gray balance the input scanner values to CIELAB L* using three 1-dimensional LUTs generated from the curves shown in Figure M-1. The data shown in Figure M-2 shows that the RGB channels are linear with respect to L* after the gray balancing.
5. Normalize the gray balanced scanner DCs between 0 and 1 by the minimum and the maximum.

6. Regress the normalized gray-balanced DC versus $L^*a^*b^*$ to get a 3x14 rotation matrix.

$$\text{LAB} = \text{M}^*\text{RGB}$$

where $\text{LAB}$ is a 3xN matrix of measured $L^*a^*b^*$ values, $\text{M}$ is a 3x14 matrix, $\text{RGB}$ is a 14xN matrix of gray-balanced digital counts consisting of $(R, G, B, RG, RB, R^2, G^2, B^2, RGB, R^3, G^3, B^3, \text{constant})$. $\text{M}$ is determined using a least squares solution.

7. Calculate $\Delta E_{ab^*}$ errors between measured and predicted colors, Figure M-3.
Figure M-1 Scanner RGBs versus L* for the neutral ramp.

Figure M-2. Gray balanced scanner RGBs versus L* for the neutral ramp.
Figure M-3. Histogram of $\Delta E_{ab}^*$ values of the models prediction of the 5x5x5 dependent data.

Model Testing

1. Print 25 random patches.

2. Measure spectral reflectance factors using 45/0 geometry.


4. Normalize XYZs according to step #2 above.

5. Gray balance scanner RGBs.
6. Normalize Scanner RGBs.

7. Generate predicted L*a*b* by using 3x14 prediction matrix from step #6 above.

8. Calculate $\Delta E_{ab}^*$ between the measured and the model predicted CIELAB values. A histogram of these errors is given in Figure M-4.

---

Figure M-4. Histogram of prediction errors for the scanner model for an independent data set.
Appendix N – Illuminant Metamerism for Print-to-Print Experiments

In order to calculate the effects of the light source metamerism for the print-to-print experiments a series of spectral reflectances from the Xpress and MajestiK printers were obtained. These reflectances were converted to CIELAB coordinates using the 10nm ASTM D50 spectral weights for the 1931 2-degree observer and the spectral radiance of the actual conditions the prints were viewed under. In both cases the colorimetry was normalized to a perfect-reflecting diffuser not the paper white. This was done to show the effects of changing the spectral power distribution of the light source. If the colorimetry was normalized to the tristimulus values of the paper some of the shifts may be masked. Error vector plots are shown in Figure N-1 and N-2 for the two printers. The base of the arrows represents the Illuminant D50 \([a^*, b^*]\) values and the head of the arrows represent the source D50 \([a^*, b^*]\) values.

These errors indicate the predominant color shifts happen in the blue region of color space. For both printers the “blue” colors shift toward magenta/red. In this region of color space the shifts are more in hue than in chroma. (Note: in the “yellow” region of color space the shifts are more in chroma.) Therefore, reddish “color-balance” shifts in the prints maybe expected.
The colorimetric shifts for the MajesyiK printer were a mean $\Delta E_{ab}^* = 1.7$ and a maximum $\Delta E_{ab}^* = 7.8$. The colorimetric shifts for the Xpress printer were a mean $\Delta E_{ab}^* = 1.62$ and a maximum $\Delta E_{ab}^* = 6.0$. (Note: Histograms of $\Delta E_{ab}^*$ differences for both printers are shown in Figure N-3 and N-4.)

Figure N-1. Illuminant metamerism effects for the MajesyiK printer.
Figure N-2. Illuminant metamerism effects for the Xpress printer.
Figure N-3. Histogram of colorimetric shifts between Illuminant D50 and the florescent D50 source used in the print experiments, for the MajestiK printer.

Figure N-4. Histogram of colorimetric shifts between Illuminant D50 and the florescent D50 source used in the print experiments, for the Xpress printer.
Appendix O – Print-to-Print Pilot Experiment Images

The images shown in Figure O-1 – O-2 were used in the print-to-print pilot experiment performed in Section 5.

Figure O-1a,b. Couple and Deer images.
Figure O-2a,b. Hats and Mt-Fuji images.
Appendix P – Print-to-Print Pilot Experiment Interval Scales

The interval scales shown in Figures P-1 – P-8 are from the print-to-print pilot experiment performed in Section 5.

Figure P-1. Interval scale for the Couple image printed on the Xpress printer. From print-to-print pilot experiment performed in Section 5.
Figure P-2. Interval scale for the Couple image printed on the MajestiK printer. From print-to-print pilot experiment performed in Section 5.

Figure P-3. Interval scale for the Hats image printed on the Xpress printer. From print-to-print pilot experiment performed in Section 5.
Figure P-4. Interval scale for the Hats image printed on the MajestiK printer. From print-to-print pilot experiment performed in Section 5.

Figure P-5. Interval scale for the Mt-Fuji image printed on the Xpress printer. From print-to-print pilot experiment performed in Section 5.
Figure P-6. Interval scale for the Mt-Fuji image printed on the MajestK printer. From print-to-print pilot experiment performed in Section 5.

Figure P-7. Interval scale for the Deer image printed on the Xpress printer. From print-to-print pilot experiment performed in Section 5.
Figure P-8. Interval scale for the Deer image printed on the MajestiK printer. From print-to-print pilot experiment performed in Section 5.
Appendix Q – Fujix Pictrography 3000 Characterization

The characterization process for the Fujix Pictrography 3000 printer consisted of printing a 10x10x10 uniform sampling of digital counts and measuring their spectral reflectance factor using a Gretag XPM60 Spectrophotometer using 45/0 geometry. These spectral reflectance factors were converted to CIELAB values using the ASTM D50 spectral weights. The colorimetry was normalized to the paper white. These 1000 CIELAB points represent the output nodes of an RGB-to-CIELAB LUT. Therefore, RGB values were converted to CIELAB using tetrahedral interpolation (Hung (1993)).

In order to test the accuracy of the device characterization, 25 random RGB values were generated. These RGB values were printed on the Fujix Pictrography 3000 printer. The spectral reflectance factors of the samples were measured and converted to CIELAB values (LAB_Measured) in the manner described above. The random RGB digital counts were converted to CIELAB values (LAB_Estimated) using the device characterization LUT and tetrahedral interpolation. The colorimetric errors are summarized by a mean $\Delta_{E_{ab}}^* = 2.1$ and a maximum $\Delta_{E_{ab}}^* = 3.9$. The histogram in Figure Q-1 shows the distribution of the colorimetric errors between the measured (LAB_Measured) and the estimated (LAB_Estimated) CIELAB values.
Figure Q-1. Histogram of $\Delta E_{ab}$ errors between the measured (LAB_Measured) and the estimated (LAB_Estimated) 25 random patches.
Appendix R – Print-to-Print Experiment Images

The image shown in Figure R-1 – R-4 were used in the print-to-print soft-proofing experiment performed in Section 5.

Figure R-1a,b. Daffodil and Family-on-Blanket images.

Figure R-2a,b. Grapes and Leaf images.
Figure R-3a,b,c. Horses-in-Mist, Orthodox-Temple, and Rose images.

Figure R-4a,b. Sunset and Undersea-Lights images.
Appendix S – Soft-Proofing Experiment Interval Scales

The interval scales shown in Figure S-1 – S-9 are for the individual images tested in the soft-proofing experiment performed in Section 5.

Figure S-1. Interval scale for the Daffodil image from the soft-proofing experiment performed in Section 5.
Figure S-2. Interval scale for the Family-on-Blanket image from the soft-proofing experiment performed in Section 5.

Figure S-3. Interval scale for the Grapes image from the soft-proofing experiment performed in Section 5.
Figure S-4. Interval scale for the Horses-in-Mist image from the soft-proofing experiment performed in Section 5.

Figure S-5. Interval scale for the Leaf image from the soft-proofing experiment performed in Section 5.
Figure S-6. Interval scale for the Orthodox-temple image from the soft-proofing experiment performed in Section 5.

Figure S-7. Interval scale for the Rose image from the soft-proofing experiment performed in Section 5.
Figure S-8. Interval scale for the Sunset-Light-House image from the soft-proofing experiment performed in Section 5.

Figure S-9. Interval scale for the Under-Sea-Lights image from the soft-proofing experiment performed in Section 5.
Appendix T - Inverse Sigmoidal-Lightness Expansion Functions

The following process was used to generate the inverse sigmoidal-remapping functions used in the gamut-expansion experiments in Section 6.

**LUT Inversion:**

- Generate a 12 bit forward LUT, \( S \), using the sigmoidal-contrast model using the black-point of the current source device and the normal-lightness class. For example, suppose the \( L^*_{\text{spec(min)}} = 22.4 \).

\[
\begin{array}{ccccccccc}
0 & 1 & 2 & 3 & \ldots & 4094 & 4095 \\
S &=& 22.4 & 22.7 & 23.9 & 26.8 & \ldots & 99.6 & 100
\end{array}
\]

To use vector \( S \) as a LUT, the input \( L^* \) values, \( L^*_{\text{in}} \), are converted to 12 bit integers by \( \text{ROUND}(L^*_{\text{in}}/100^*(2^{12}-1)) \). The output lightness values, \( L^*_{\text{out}} \), are given by \( L^*_{\text{out}}=S( \text{ROUND}(L^*_{\text{in}}/100^*(2^{12}-1)) ) \). The integer scaled lightness values, \( \text{ROUND}(L^*_{\text{in}}/100^*(2^{12}-1)) \), are used as index values into the LUT. (Note: The first and last elements of the \( S \) correspond to input lightness values of 0 and 100 respectively.)
• Generate a 12 bit vector, \( K \), of integers numbers that go from \([0 \ 2^{12}-1]\). These values represent pseudo lightness values, on the range of \([0 \ 100]\), scaled into 12 bit integer values from \([0 \ 2^{12}-1]\).

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & \ldots & 4094 & 4095 \\
\end{array}
\]

\[
K = \begin{pmatrix}
0 & 1 & 2 & 3 & \ldots & 4094 & 4095 \\
\end{pmatrix}
\]

• Define the source device black point in terms of a 12 bit integer. Therefore, \( L^{*}_{src(min)} \) gets converted to the nearest 12 bit integer by \( \text{ROUND}(L^{*}_{src(min)}/100*(2^{12}-1)) \).

• For values in \( K \) less than \( \text{ROUND}(L^{*}_{src(min)}/100*(2^{12}-1)) \), set the points in inverse LUT, \( S^{-1} \), to \( L^{*}_{dest(min)} \). In the example shown below the \( L^{*}_{dest(min)} = 0 \).

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & \ldots & \text{ROUND} & \ldots & 4094 & 4095 \\
\end{array}
\]

\[
S^{-1} = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & \ldots & 4094 & 4095 \\
\end{pmatrix}
\]

• For values in \( K \) greater than \( \text{ROUND}(L^{*}_{src(min)}/100*(2^{12}-1)) \), perform linear interpolation to get their corresponding inverse sigmoid values. The linear interpolation function used has the form:

\[
y' = \text{interp}(y, x, x'), \quad \text{(T-1)}
\]

where \( y \) and \( x \) are the dependent and independent variables in the function. The \( x' \) values are a set of new independent values that are used
to create estimates, \( y' \), of the input \( (x, y) \) data. Therefore the output LUT, \( S^{-1} \), for all values of \( K \) greater than or equal to the source black point, is given by:

\[
S^{-1} = \text{interp}(K, S, K). \quad (T-2)
\]

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & \ldots & \text{ROUND} & \ldots & 4094 & 4095 \\
(L*\text{src(min)}/100*4095) & & & & & & & & \\
S^{-1} &=& 0 & 0 & 0 & 0 & 0 & 0 & \ldots & 99.6 & 100 \\
\end{array}
\]
Appendix U – Gamut Expansion Images

The images shown in Figures U-1 – U-4 were used in the gamut-expansion experiments performed in Section 6.

Figure U-1a,b. Daffodil image and Eiffel-Tower image.

Figure U-2a,b. Family-on-Blanket image and Grapes image.
Figure U-3a,b. Hat image and Ladies-Yellow-Wall image.

Figure U-4a,b. Leaf image and Red-Barn image.
Appendix V – Gamut-Expansion Experiment Interval Scales

The interval scales shown in Figures V-1 – V-8 are from the print-to-print pilot experiment performed in Section 6.

Figure V-1. Interval scale for the Daffodil image from the gamut-expansion experiment performed in Section 6.
Figure V-2. Interval scale for the Eiffel-Tower image from the gamut-expansion experiment performed in Section 6.

Figure V-3. Interval scale for the Family-on-Blanket image from the gamut-expansion experiment performed in Section 6.
Figure V-4. Interval scale for the Grapes image from the gamut-expansion experiment performed in Section 6.

Figure V-5. Interval scale for the Hats image from the gamut-expansion experiment performed in Section 6.
Figure V-6. Interval scale for the Hats image from the gamut-expansion experiment performed in Section 6.

Figure V-7. Interval scale for the Leaf image from the gamut-expansion experiment performed in Section 6.
Figure V-8. Interval scale for the Red-Barn image from the gamut-expansion experiment performed in Section 6.
Appendix W – Monitor-to-Print Experiment Interval Scales

The interval scales shown in Figures W-1 – W-7 are from the monitor-to-print experiment performed in Section 5.2.

Figure W-1. Interval scale for the boy-blue-sweater image from the monitor-to-print experiment performed in Section 5.2.
Figure W-2. Interval scale for family-by-car image from the monitor-to-print experiment performed in Section 5.2.

Figure W-3. Interval scale for flying-bride image from the monitor-to-print experiment performed in Section 5.2.
Figure W-4. Interval scale for kid-in-tire image from the monitor-to-print experiment performed in Section 5.2.

Figure W-5. Interval scale for ladies-yellow-wall image from the monitor-to-print experiment performed in Section 5.2.
Figure W-6. Interval scale for logger image from the monitor-to-print experiment performed in Section 5.2.

Figure W-7. Interval scale for wine-man image from the monitor-to-print experiment performed in Section 5.2.