# UBC EECE 541

# Project Report

# Colour Correction for iTMO

---

Submitted to:
**Dr. Mahsa Pourazad**


Submitted by:

| | | |
|---|---|---|
| **An, Siqi** | **97950166** | **anantjjz@gmail.com** |
| **Chan, Benny (Chun Yin)** | **42378075** | **bennyccy@gmail.com** |
| **Liu, Liming** | **95660163** | **shannonsarah2333@gmail.com** |
| **Wang, Zhen** | **98552169** | **wangzhen5161@outlook.com** |


Submitted on:
**2017-12-15**

# Table of Contents

**Table of Contents** 2

**Acknowledgement** 4

**Abstract** 5

**Introduction** 6

**Colour Space and Color Space Conversion** 7
- RGB Color Space 7
- XYZ Color Space 7
- Lab Color Space 7
- RGB to XYZ Color Space Conversion 8
- XYZ to Lab Color Space Conversion 8

**Existing Work - iTMO & Colour Shifting** 9

**Detailed iTMO Process & Implementation** 12

**Visualizing HDR Images** 13

**DeltaE** 14

**Colour Correction Adjustment** 14

**Color Correction Adjustment with Global Alpha** 16

**Colour Correction Adjustment with Local Alpha** 20

**Visualization of Alpha Map** 22

**Alpha Maps 3D & 2D Graphs** 24
- Alpha Map 3D Graphs 24
  - Bist Alpha Map 3D Graphs 24
  - Akyuz Alpha Map 3D Graphs 28
- Alpha Map 2D Graphs 33
  - Bist Alpha Map 2D Graphs 33
  - Akyuz Alpha Map 2D Graphs 35
- Alpha Map Histograms 37
  - Bist Alpha Map Histogram 37
  - Akyuz Alpha Map Histogram 38

**Colour Correction Results** 39

Colour Correction Results for Bist iTMO HDR Images 39
DeltaE Values 43
Issue with Colour Correction Adjustment with Bist Local Alpha Map 44
Range of A & B Values 46
Colour Correction Results for Akyuz iTMO HDR Images 48
DeltaE Values 51
Range of A & B Values 52

**Conclusion 53**

**Reference 54**

**Appendix 1 - Seven test images for iTMO 55**

**Appendix 2 - Matlab Code for iTMOs 58**

**Appendix 3 - Matlab Code for Euclidean Distance Analysis 59**

**Appendix 4 - Matlab Code for Alpha Map Visualization 60**

**Appendix 5 - Matlab Code for Color Region Determination 74**

**Appendix 6 - Matlab Code for DeltaE 74**

**Appendix 7 - Matlab Code for Main function of global alpha color correction method 75**

**Appendix 8 - Matlab Code for Main function of local alpha color correction method 76**

**Appendix 9 - Matlab Code for alphaMap generation 78**

**Appendix 10 - Matlab Code for finding the alpha corresponding to the minimum deltaE 80**

**Appendix 11 - Matlab Code for searching for the alpha in the alphaMap for images 81**

**Appendix 12 - Matlab Code for drawing histogram based on the alphamap 82**

**Appendix 13 - Matlab Code for comparing the bound of A and B 83**

**Appendix 14 - Additional figures of comparison between A & B 85**

## Acknowledgement

We would like to express our special thanks and gratitude to Dr. Mahsa Pourazad, Pedram Mohammadi, Dr. Panos Nasiopoulos, Maryam Azimi, and Stelios Ploumis for provideing us with guidance and help on the project.

## Abstract

With the development of high dynamic range (HDR) content generation and display technologies, various tone mapping operators (TMO) and Inverse tone mapping operators (iTMO) are proposed in the past years to convert HDR content for SDR displace and vice versa respectively [1].

iTMOs are able to produce HDR content that are acceptable to content viewer without comparing to the original SDR content. However, the conversion of content from SDR to HDR using iTMO introduces colour shifting in the resulting HDR content. This colour shift may contribute to a loss of artistic intent in the SDR content.

In this project, colour correction adjustment is applied on iTMO resulting HDR images by scaling the chrominance information to alter the colour saturation.

# Introduction

In recent years, the need for High dynamic range (HDR) image, which is able to display content with increased luminance range, increases rapidly in various fields. In order to convert between high dynamic range content and standard dynamic range (SDR) content, inverse Tone Mapping  Operator(iTMO) is employed.

In a conversion process, an iTMO may cause color shifting problem due to the expansion of luminance and chrominance. So the colour perceived in the HDR content is different than that in the original SDR content.

In order to deal with the color shifting problem, we first focus on the color shifting problem in 4 iTMOs, and then mitigate the colour shifting in the HDR images by using an alpha parameter to perform colour correction adjustment to the HDR images. We experiment on both global alpha and local alpha, and use deltaE and subjective test? to compare the result. For the local alpha, we also try to find regularity in the trend for alpha by visualizing alpha map.

In this proposal, we will first introduce detailed processes of 4 iTMO and the color space conversion related to our work. Then, basic concepts of deltaE and HDR image visualization are covered. After that the process for performing color correction with global alpha and local alpha as well as the visualization of local alpha map are presented. Lastly, the comparison for different alphas and the results for color correction are discussed.

# Colour Space and Color Space Conversion

Color space is a specific organization of colors, which allows for reproducible representations of colors in both analog and digital representations, and each color is represented by a single dot in each system. The following part will give a brief introduction on RGB color space, LAB color space, XYZ color space, and the conversion between them, which are used in our work.

## RGB Color Space

An RGB color space is any additive color space based on RGB color model [4] (an additive color mode where red, green, blue combine randomly to reproduce a broad array of colors). RGB color space is defined by three color red, green, and blue. Any chromaticity can be produced with the combination of those three color [10].

## XYZ Color Space

XYZ color space is one of the first mathematically defined perceptual color space called also CIE 1931 XYZ. It was created by International Commission on Illumination (CIE) ,and derived from a series of experiments done by W. David Wright and John Guild [11]. CIE 1931 XYZ color space contains all color sensations that are visible to a person with average eyesight, that's the reason why CIE XYZ (Tristimulus values) is a device-invariant representation of color.

## Lab Color Space

The Lab color space is a color space specified by the International Commission on Illumination. It describes mathematically all perceivable colors in the three dimensions L for lightness and a and b for the color opponents green–red and blue–yellow. The terminology "Lab" originates from the Hunter 1948 color space[12].

## RGB to XYZ Color Space Conversion

RGB space to XYZ space conversion is a linear transformation. The standardized transformation settled upon by the CIE special commission is as shown below [13]. The numbers in the conversion matrix below are exact, with the number of digits specified in CIE standards.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{0.17697} \begin{bmatrix} 0.49000 & 0.31000 & 0.20000 \\ 0.17697 & 0.81240 & 0.010630 \\ 0.0000 & 0.01000 & 0.99000 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

## XYZ to Lab Color Space Conversion

Lab color space can't be converted from RGB space directly and so we first convert RGB to XYZ and then convert XYZ to Lab using the equations below[14].

$$L = 116 f(\tfrac{Y}{Yn}) - 16$$

$$a = 500(f(\tfrac{X}{Xn}) - f(\tfrac{Y}{Yn}))$$

$$b = 200(f(\tfrac{Y}{Yn}) - f(\tfrac{Z}{Zn}))$$

, where

$$f(t) = \sqrt[3]{t} \; if \; t > (\tfrac{36}{841})$$

$$f(t) = \tfrac{t}{1225} + \tfrac{4}{29} \; otherwise$$

$$Xn = 95.047$$

$$Yn = 100.000$$

$$Zn = 108.883$$

## Existing Work - iTMO & Colour Shifting

In our project, four iTMO methods are used to serve as the starting point. They are Akyuz [2], Banterle [3], Bist [1]& Meylan [4]. In general, these iTMO methods expand the RGB values of the SDR content based on the difference in luminance between the SDR content and target level. The target level is determined by the output luminance range of the HDR display. Figure 1 shows a generic iTMO process.

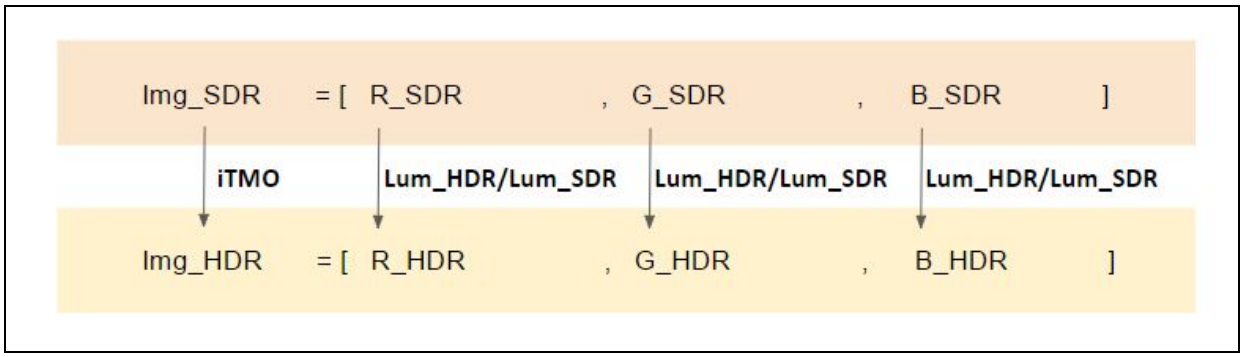



Figure 1: General iTMO Process

To extract luminance information from RGB values, the iTMOs use these different methods:

Bist first converts RGB to XYZ by using the matrix

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{0.17697} \begin{bmatrix} 0.49000 & 0.31000 & 0.20000 \\ 0.17697 & 0.81240 & 0.010630 \\ 0.0000 & 0.01000 & 0.99000 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

and then use the value of Y as the luminance [1].

Akyuz, Banterle, and Meylan directly extract luminance from RGB by using

$$Luminance = 0.2126 * R + 0.7152 * G + 0.0722 * B \qquad [2], [3], [4]$$

More specifically, iTMOs employ different techniques to convert images from SDR to HDR range.

Banterle separates the processing into two steps. One is to do ITMO so as to get the expanded the original SDRI to HDRI and the other is to use expanded map to deal with the area with high luminance. Afterwards, interpolate these two into a final version of HDRI image.

Bist uses a user study based equation of $\Upsilon$ to change the luminance of SDR image[1]:

$$\gamma = 1 + \log_{10}(\frac{1}{\overline{L^*}})$$

$\overline{L^*} = L^*_{median}/100$ and has been clipped in the interval [0.05(:)0.95]

$L^*_{median}$ is the median of the lightness channel of each frame in the CIE-Lab domain

$$Y_{HDR} = L_{MAX} \times Y^{\gamma}{}_{SDR}$$

,where

$Y_{HDR}$ is the Y component of XYZ colorspace in HDR domain

$Y_{SDR}$ is the Y component of XYZ colorspace in SDR domain

$L_{MAX}$ is the maximum display luminance (1000 nits)

Then, RGB values in HDR domain is calculated by the ratio of luminance for HDR and SDR:

$$\begin{bmatrix} R_{HDR} \\ G_{HDR} \\ B_{HDR} \end{bmatrix} = \frac{Y_{HDR}}{Y_{SDR}} \begin{bmatrix} R_{SDR} \\ G_{SDR} \\ B_{SDR} \end{bmatrix}$$

Meylan method uses a method to automatically segment the input image into its diffuse and specular components, and then use the result of the segmentation to construct the tone scale, which is applied to each pixel of the image [4].

Some of these iTMOs are able to produce HDR content that are acceptable to content viewer. However, when compared to the original SDR content, it was observed that the colour perceived in the HDR content is different than that in the original SDR content.

To produce the HDR content, the iTMO scales the RGB values of the SDR content according to the luminance level of the SDR content and target luminance level. This scaling generally reduce the saturation of the colour appears in the resulting HDR image. The regions of A & B values occupy is reduced after the iTMO.

As region of A & B values represents the chrominance of the content, the reduction in the region after iTMO introduces colour shifting in the resulting HDR content. This colour shift may contribute to a loss of artistic intent in the SDR content.

Our project addresses this colour shifting issue by performing colour correction adjustment on resulting HDR content from iTMO. More details of the colour correction adjustment will be given the in following sections.

## Detailed iTMO Process & Implementation

This section displays the generic process of iTMO with more implementation details. This discussion provides more context of a generic iTMO process excluding the specific techniques used in an iTMO. The context would help explain how our colour correction adjustment fit into a generic iTMO implementation.

The generic iTMO process includes the steps as listed below.

1. Normalizes the RGB values of the images ranging 0-255 to ranging 0-1.
2. Remove the gamma that is applied on the image.
3. iTMO uses their own techniques to find the luminance ratio that scales each of the RGB values of the image.
4. The SDR RGB values are then scaled to the HDR display luminance value. Gamma is not applied on the output HDR images.
5. Depending on how the HDR images are to be displayed, .png and .hdr format files are created from the HDR image's RGB values.

In figure 2, the generic iTMO process described above is illustrated.

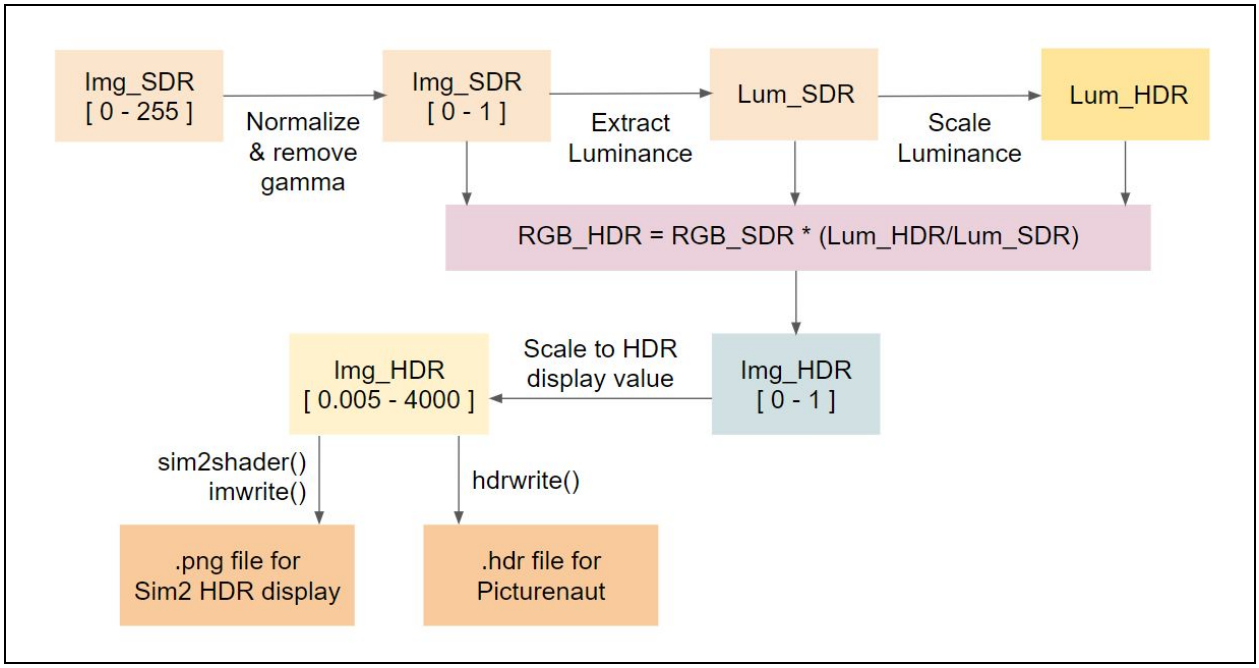



Figure 2: Process chart with iTMO producing HDR image.

## Visualizing HDR Images

The Sim2 HDR display is used in this project. It has a display luminance range of 0.005 to 4000 nits or cd/$m^2$. So, the normalized RGB values of the HDR image are scaled to the range of 0.005 to 4000. Additionally, a shader function convert the HDR content ranging [0.005-4000] to [0-255] to be used by the imwrite() function[5] to create the .png file for the Sim2 HDR display.

Alternatively, an hdrwrite() function is used to convert the RGB values of the HDR image ranging [0.005-4000] into a .hdr file[6]. The .hdr file can be viewed by Picturenaut, a software that allows viewer to view HDR images on SDR display by varying the exposure setting.

The Sim2 HDR display and Picturenaut software are the two means to perform subjective observations on the HDR images. Other than the subjective observation of the HDR images, a mathematical metric is also used in this project to represent the perceived colour difference between the original SDR image and the HDR images. More about this metric is discussed in the following section.

## DeltaE

DeltaE is used in this project as an objective metric to estimate the perceived colour difference between two images. DeltaE is a calculation of Euclidean distance between the LAB values of two images LAB values in the LAB colour space. The lower Euclidean distance between the LAB values represents a closer perceived colour resemblance between two images. The equation to calculate deltaE is as shown below [7]:

$$\text{deltaE} = \sqrt{\sum[(L_1 - L_2)^2 + (A_1 - A_2)^2 + (B_1 - B_2)^2}$$

,where $L_1$, $A_1$, $B_1$ are the LAB values from one image and $L_2$, $A_2$, $B_2$ are the LAB values from another image.

In the project, we learned that, for valid comparison, the immediate result of the iTMO or colour correction in form of matrices needs be used for the deltaE calculation. When the resulting matrix that represents the image is saved as .hdr and loaded as a matrix, error is introduced to the calculation.

The code for the deltaE estimation is attached in appendix 5.

## Colour Correction Adjustment

To mitigate the colour shifting in the HDR images from the iTMO, we performed colour correction adjustment to the HDR images.

At the early stage of the project, we have explored ways to manipulate colour by altering the RGB values of an images. Nonetheless, doing so in the RGB would affect the luminance level. We then choose to perform the colour manipulation in LAB colour space because (1) luminance and chrominance information is isolated; (2) deltaE is calculated in LAB colour space, and (3) LAB is perceptually close to human visual system.

With that, we further define our colour correction method to alter only the chrominance A and B, and to keep the luminance L unchanged. We define a constant called alpha that is used to scale both the chrominance A & B. The scaling of both A & B with the same constant that would result in a change in colour saturation but not hue, which is desirable for the colour correction purpose.

In figure 3, the colour correction adjustment is added to overall process.

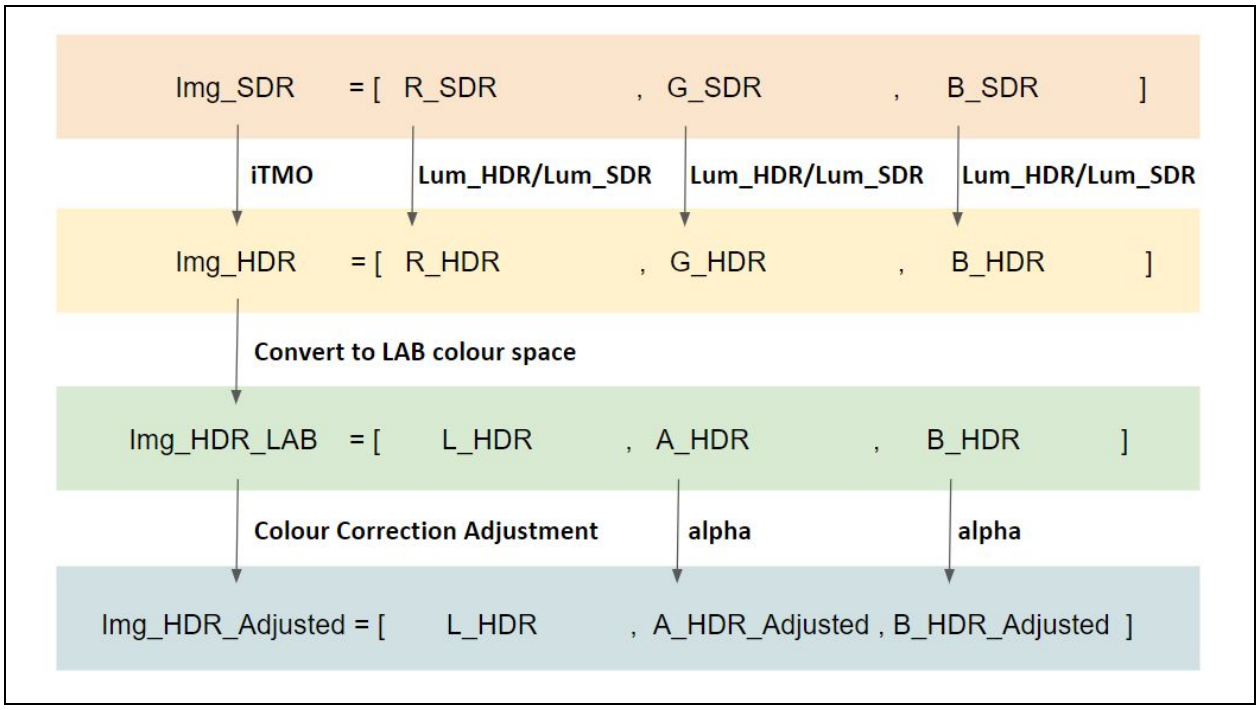



Figure 3: iTMO & colour correction process overview.

## Color Correction Adjustment with Global Alpha

The previous section establishes the method of the colour correction adjustment. This section and the next presents the implementation of the colour correction adjustment.

In order to implement the colour correction adjustment, the value of alpha is needed. To find the values of the alpha, a trial-and-error approach is used. An array of discrete numbers are applied to scale both the chrominance A & B values of an HDR image[8]. The deltaE values between each resulting HDR image and the original SDR image are calculated. The number that results in the lowest deltaE value is selected to be the global alpha value for that particular image[9].

We applied this approach to the seven test images, each image is first processed by 4 different iTMOs. As a result, the approach is repeated 28 times to find the global alpha values for each test images from each iTMO. The results for each iTMO resulting HDR images are plotted in the graphs shown in figure 4 to 7. Table 1 summarizes the results with the global alpha values and their corresponding deltaE.

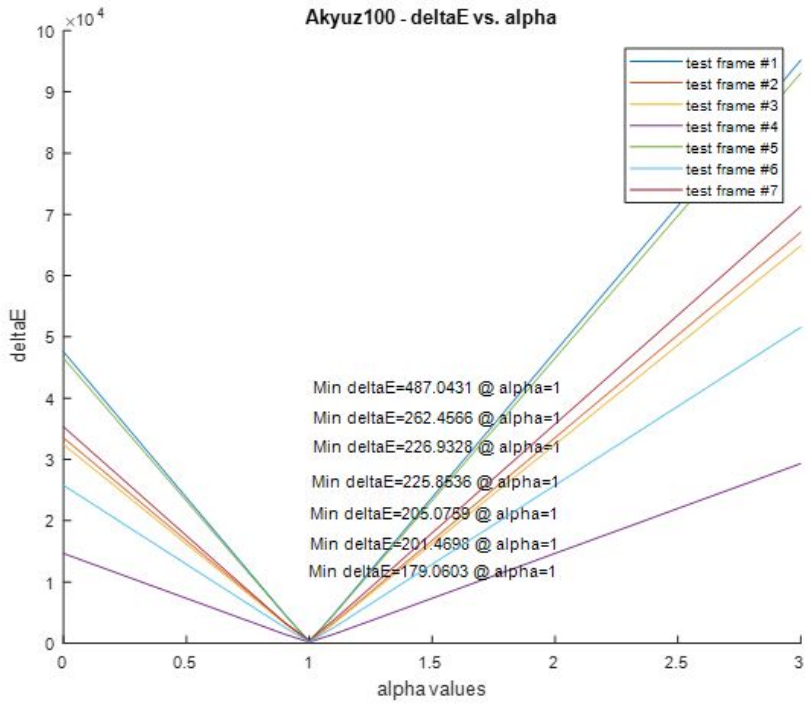



Figure 4: deltaE-alpha in Akyuz

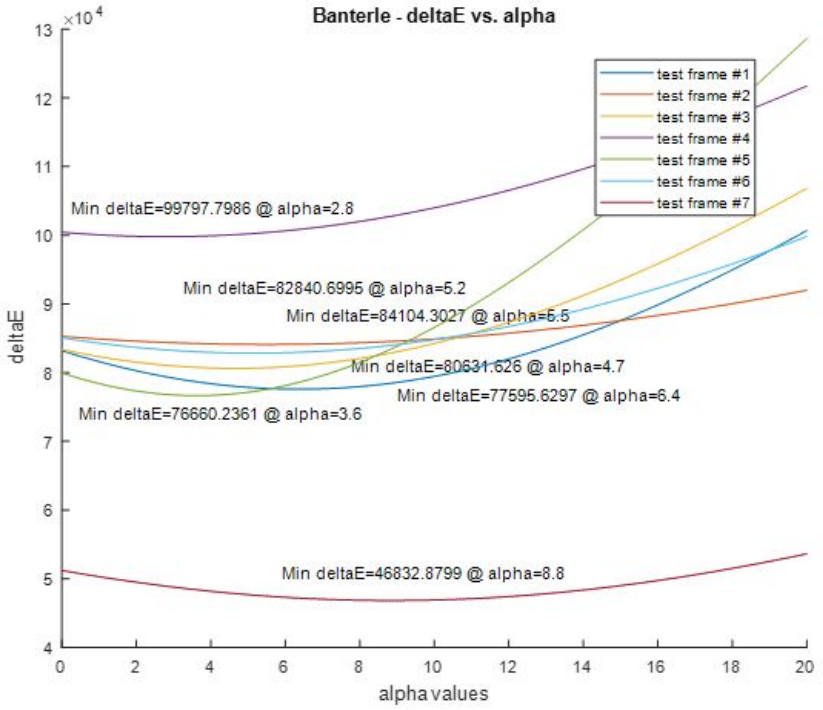



Figure 5: deltaE-alpha in Banterle

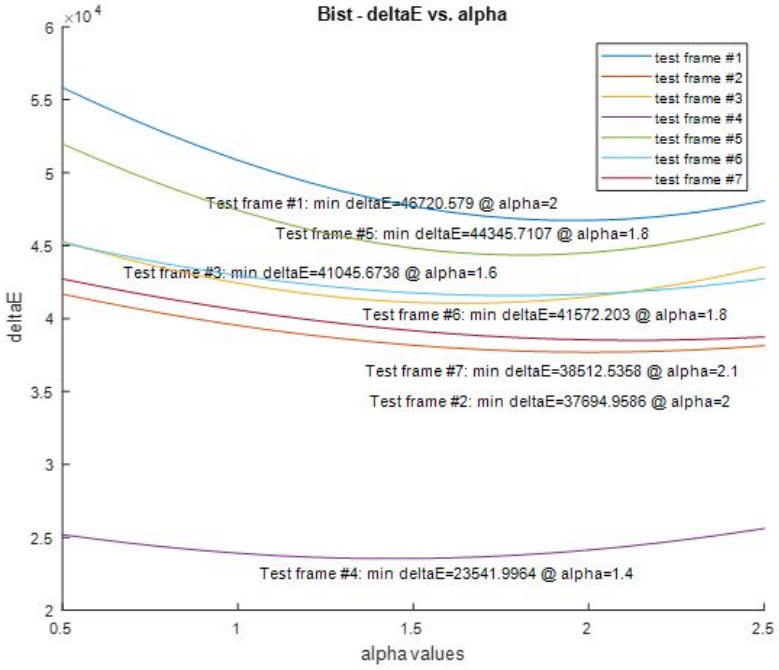



Figure 6: deltaE-alpha in Bist

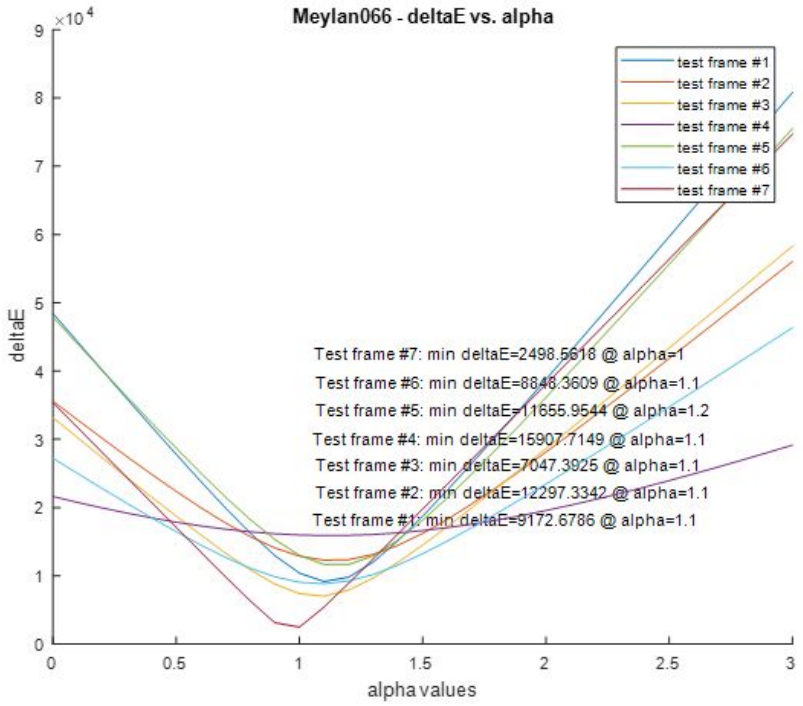



Figure 7: deltaE-alpha in Meylan

| | **Akyuz gamma = 1.0** | | **Banterle** | | **Bist** | | **Meylan p=0.66** | |
|---|---|---|---|---|---|---|---|---|
| **Test Frames** | DeltaE min | Corresponding Alpha Values | DeltaE min | Corresponding Alpha Values | DeltaE min | Corresponding Alpha Values | DeltaE min | Corresponding Alpha Values |
| 1- Chair | 262.46 | 1.00 | 77595.63 | 6.40 | 46720.58 | 2.00 | 9172.68 | 1.10 |
| 2 - Car | 226.93 | 1.00 | 84104.30 | 5.50 | 37694.96 | 2.00 | 12297.33 | 1.10 |
| 3 - Lake | 201.47 | 1.00 | 80631.63 | 4.70 | 41045.67 | 1.60 | 7047.39 | 1.10 |
| 4 - Family | 205.08 | 1.00 | 99797.80 | 2.80 | 23542.00 | 1.40 | 15907.71 | 1.10 |
| 5 - Garden | 179.06 | 1.00 | 76660.24 | 3.60 | 44345.71 | 1.80 | 11655.95 | 1.20 |
| 6 - Home | 225.85 | 1.00 | 82840.70 | 5.20 | 41572.20 | 1.80 | 8848.36 | 1.10 |
| 7 - Restaurant | 487.04 | 1.00 | 46832.88 | 8.80 | 38512.54 | 2.10 | 2498.56 | 1.00 |

Table 1: DeltaE values with their corresponding global alpha values

With the 28 global alpha values found, they are applied to their corresponding images. Results are of this colour correction adjustment is displayed in the result section. Figure 8 shows the addition of colour correction adjustment in the process chart after the iTMO.

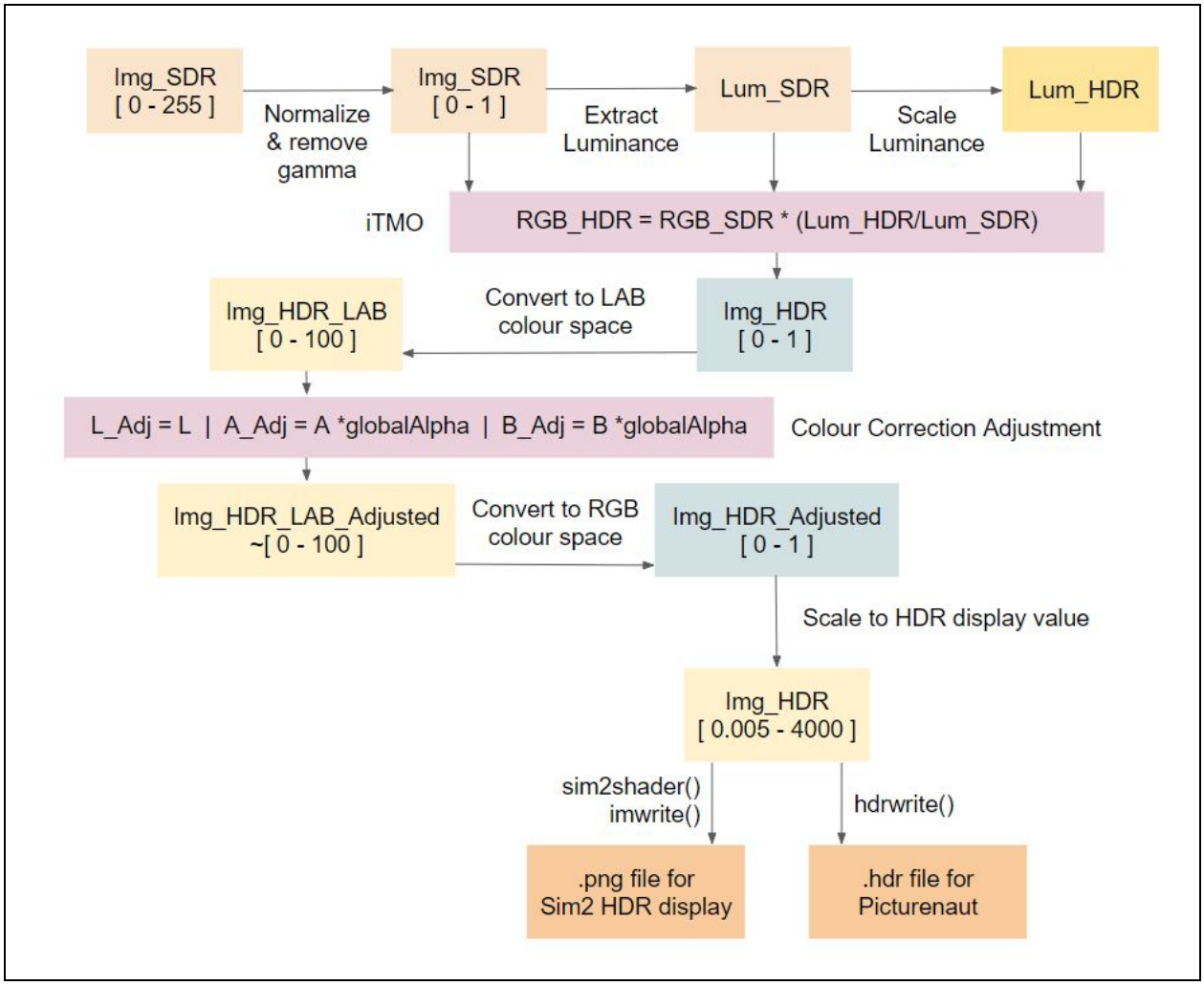



Figure 8: Process chart with colour correction adjustment using global alpha value.

## Colour Correction Adjustment with Local Alpha

In the previous section, colour correction adjustment with global alpha value was presented. In this section, the colour correction adjustment is localized to have alpha values for individual pixels of the HDR image.

With the global alpha approach, each HDR image would have one alpha value that can result in the lowest deltaE value. A single global alpha value cannot minimize the deltaE value contribution from all the LAB pixels of the HDR image. In order to reduce the deltaE value further, the local alpha approach is used. With the local alpha approach, each LAB value combination of an iTMO method has its alpha value.

The local alpha values are iTMO-dependent because each iTMO has its own technique to produce the HDR images. The local alpha values have to take into account the effect of the iTMO technique to adjust the chorminance A & B to minimize the deltaE between the adjusted HDR image and the original SDR image.

To find the local alpha values, trial-and-error approach similar to that for the global alpha value is used. The implementation is as follows:

1. A test range of LAB values is established.
2. An alpha map matrix that is indexed with LAB values for a particular iTMO is created.
3. Each of the LAB values is converted into RGB values.
4. The RGB value is processed by an iTMO to create a resulting HDR RGB value.
5. The resulting HDR RGB value is adjust with an array of discrete numbers to scale both the chrominance A & B values.
6. The number that results in the lowest deltaE value is selected to be the local alpha value for that LAB value.
7. The local alpha value is stored in the alpha map matrix.
8. Step 3 to 7 are repeated for the entire LAB value test range.

At the beginning, LAB value test ranges of L=[0:20:+100], A=[-100:20:+100], and B=[-100:20:+100] are used. 20 is the incremental interval to step through the test range. With these ranges and step, 5*10*10 = 500 LAB values are aimed to be test for each iTMO to obtain their corresponding local alpha values. However, as this approach to find the local alpha values is applied to each of the 4 iTMOs, we realized that, in general, colour correction adjustment with local alpha values is not suitable for Banterle and Meylan iTMOs because their iTMO techniques consider neighbour pixels.

More specifically, Banterle iTMO identifies areas of pixels with higher luminance than the rest of the image. Banterle iTMO uses expand map to cluster light sources near areas of high luminance. The area of high luminance received more processing and combined with the rest of the image. Essentially, Banterale iTMO considers the neighbour area of pixels to determine if more processing is needed. As a result, the colour correction adjustment with local alpha cannot be applied to Banterle resulting iTMO directly because the local alpha approach only consider individual pixel values.

Another iTMO that cannot be used with colour correction with local alpha is Meylan iTMO. Melan iTMO separates an image into specular highlight and diffused parts. The luminance of the two parts are processed differently to create the resulting HDR image. Since the local alpha approach only consider individual pixel values, it cannot be applied to Meylan iTMO.

In figure 9, the local alpha colour correction replace the global alpha approach in the process chart.

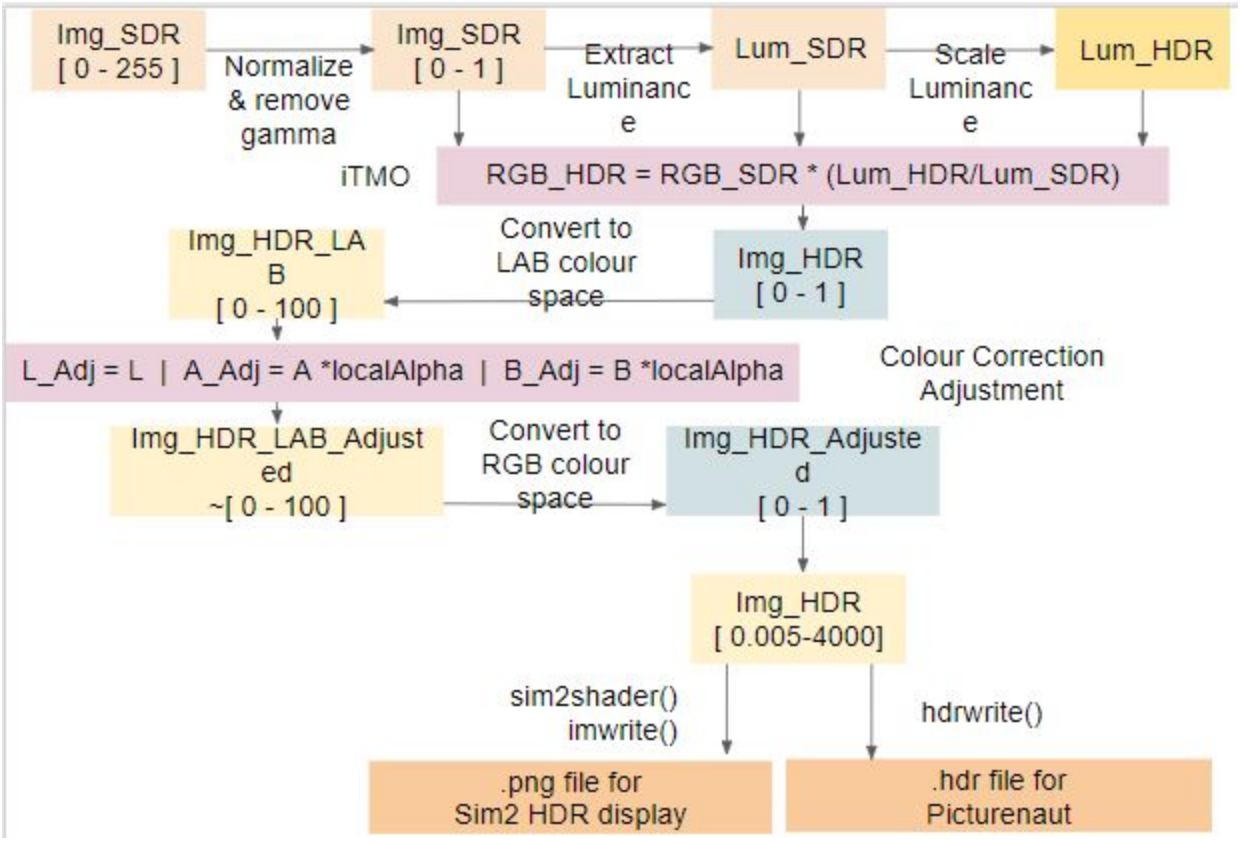



Figure 9: Process chart with colour correction adjustment using local alpha value.

## Visualization of Alpha Map

With the alpha maps constructed for Akyuz and Bist iTMOs, we created a graphical user interface (GUI) in MATLAB to visualize the alpha maps. The goal of the GUI is to identify any trend and abnormalities in the alpha map.

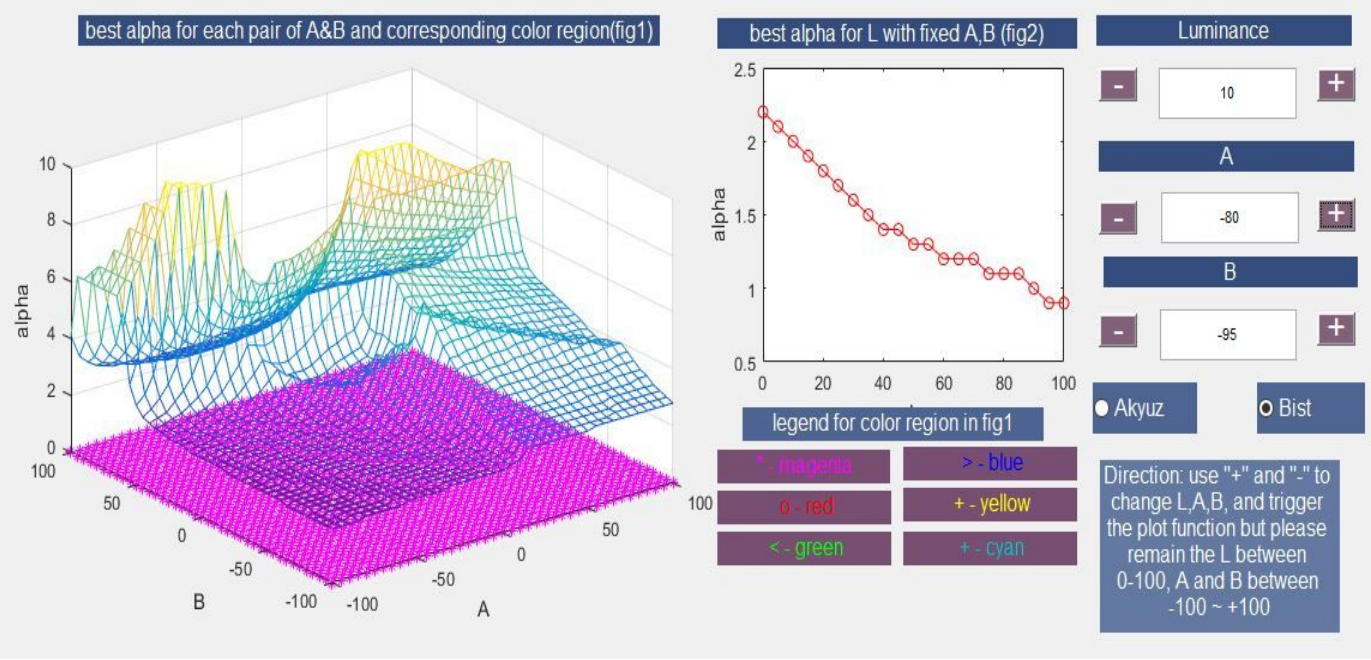



Figure 10: Graphic user interface for alpha map visualization.

Figure 10 shows the GUI for alpha map visualization. The 3D graph on the left displays data from selected iTMO alpha map in form of a surface that is defined by the alpha values at a selected luminance level across the A & B values. The iTMO and luminance level can be selected on the control panel on the right.

The markers on the surface of alpha=0 in the 3D graph indicates the colour region correspond to the LAB values. The legend of the colour region markers is shown at the bottom centre.

The 2D graph at the centre shows the relationship of alpha against luminance for a set of selected A and B values. The A & B values can be selected on the control panel on the right. In figure 10, alpha values is plotted against luminance values at A=-80 and B=-95.

The colour region in the 3D is a piece of unpublished work from Pedram Mohammadi. His work defines 6 colour regions in the YCbCr colour space using ranges of hue angle.

In YCbCr, the color regions can be determined by the angle for hue. Different hue is calculated by the equation below:

$$\text{Hue} = \Theta = tan{-}1\ (\frac{Cr}{Cb})$$

,where Cr is the red-difference chroma components, Cb is the blue-difference chroma components.

In his work, the colour regions and hue angles ranges are defined as follows:

| | | | |
|---|---|---|---|
| Magenta: | $22^\circ$ | ≤ Θ ≤ | $80^\circ$ |
| Red: | $80^\circ$ | ≤ Θ ≤ | $138^\circ$ |
| Yellow: | $138^\circ$ | ≤ Θ ≤ | $202^\circ$ |
| Green: | $202^\circ$ | ≤ Θ ≤ | $259^\circ$ |
| Cyan: | $259^\circ$ | ≤ Θ ≤ | $319^\circ$ |
| Blue: | $319^\circ$ | ≤ Θ ≤ | $22^\circ$ |

In the 3D graph of our GUI for alpha map visualization, the LAB values are converted to YCbCr values to determine its corresponding colour region with the definition above. The purpose of including the colour region in the 3D graph is to observe any relationship between the alpha values in the different colour region.

# Alpha Maps 3D & 2D Graphs

Using the GUI presented in the previous section, local alpha values for different LAB values for Akyuz and Bist iTMO are displayed as 3D graphs and 2D graphs as documented below.

## Alpha Map 3D Graphs

Each 3D graph shows the local alpha values at a luminance level. Z-axis is the alpha value; x-axis is the A value; and y-axis is the B value. The colors of the markers on the alpha=0 surface represent the color region of the corresponding LAB values.

### Bist Alpha Map 3D Graphs

The local alpha values for Bist iTMO are shown in figures 11 to 31. From the Bist alpha map 3D graphs, we observed that as luminance increases, the alpha values decreases.

In the 3D graphs of L=60 to 80, an area of reduced alpha values can be seen on the color region boundary between blue and magenta. However, limited by the timeline of the project, these observations are not investigated further.

Additionally, as the value of luminance increases from 0 to 100, the local alpha value generally reduces from 10 to 1. We suspect a correlation between the local alpha values and the luminance level. However, this is not investigated in our project and suggested as part of the future work.

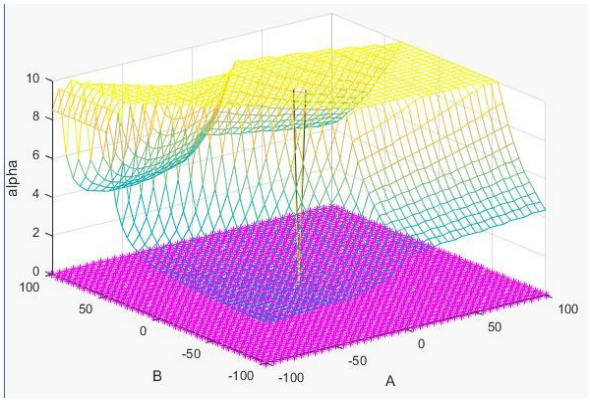



Figure 11: Bist local alpha values @ L=0.

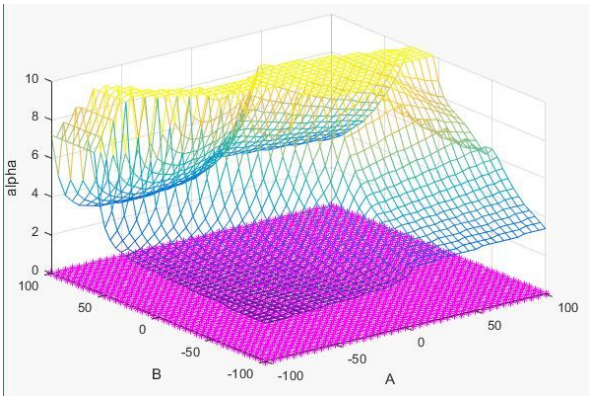



Figure 12: Bist local alpha values @ L=5

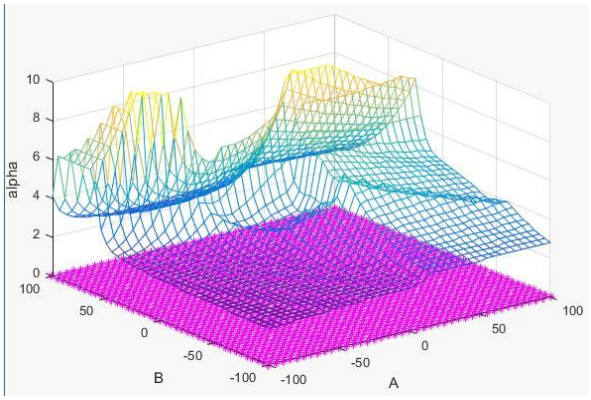



Figure 13: Bist local alpha values @ L=10

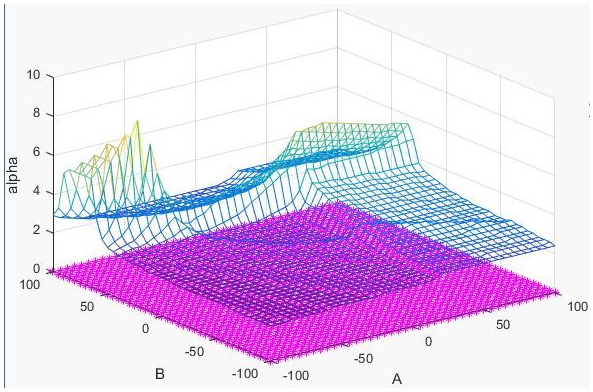



Figure 14: Bist local alpha values @ L=15

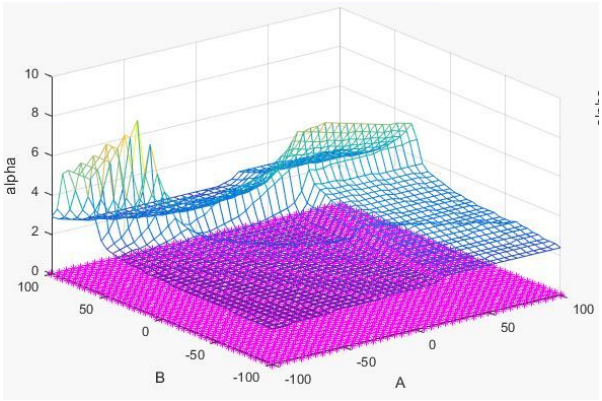



Figure 15: Bist local alpha values @ L=20

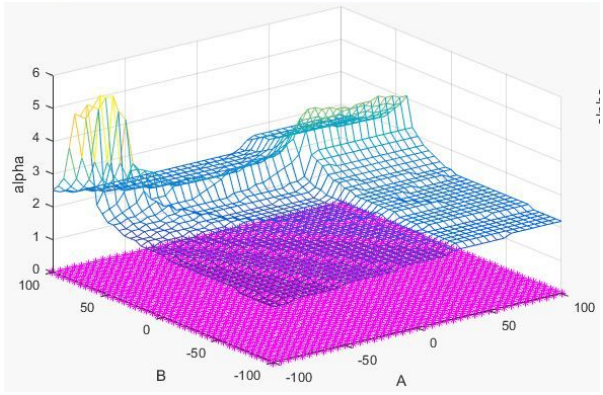



Figure 16: Bist local alpha values @ L=25

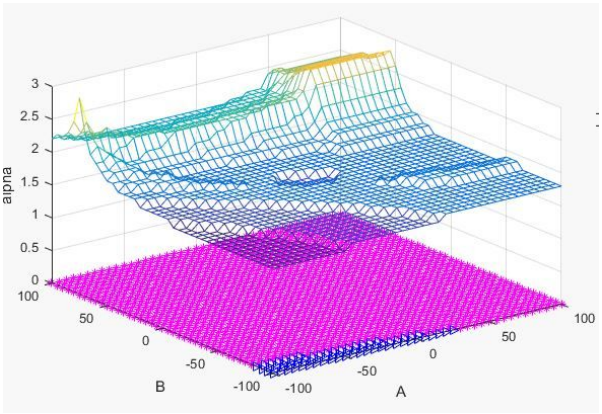



Figure 17: Bist local alpha values @ L=30

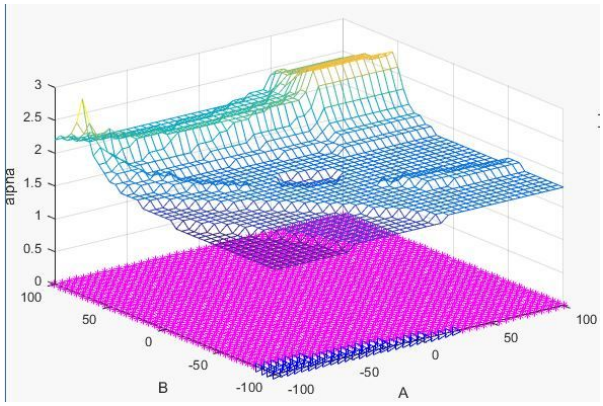



Figure 18: Bist local alpha values @ L=35

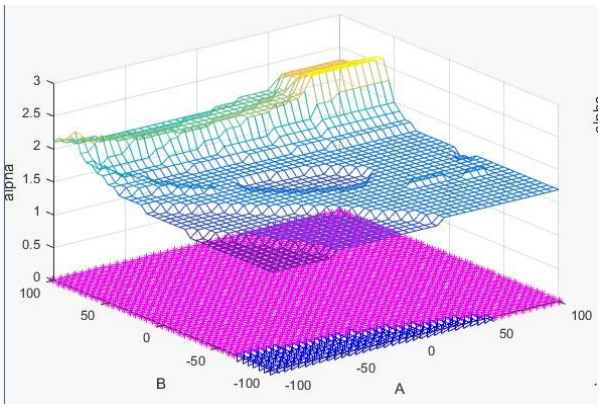



Figure 19: Bist local alpha values @ L=40

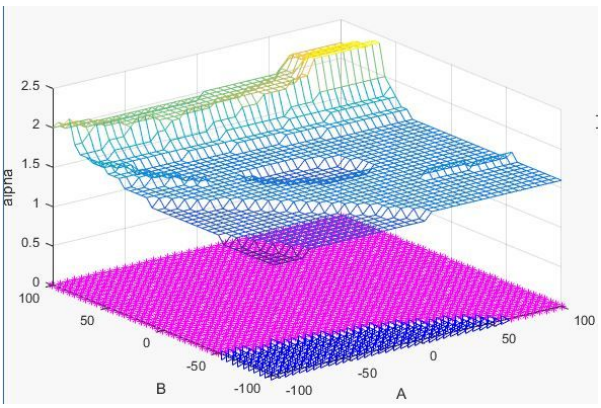



Figure 20: Bist local alpha values @ L=45

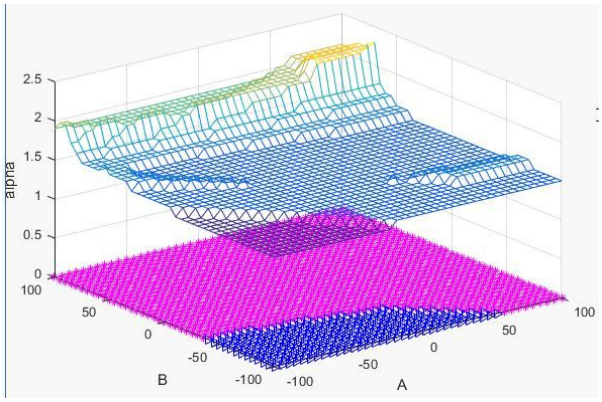



Figure 21: Bist local alpha values @ L=50

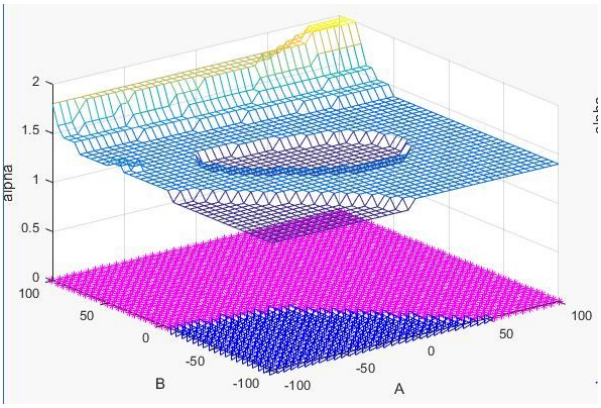



Figure 22: Bist local alpha values @ L=55

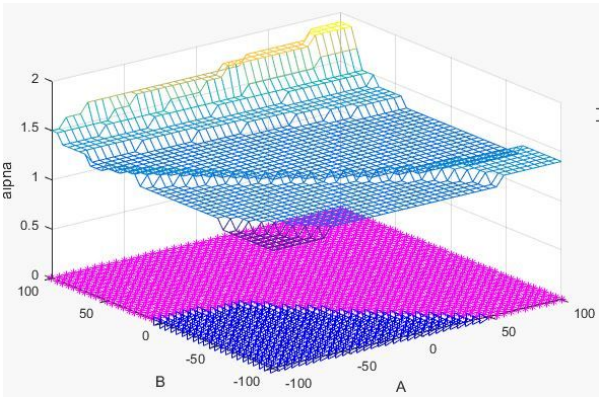



Figure 23: Bist local alpha values @ L=60

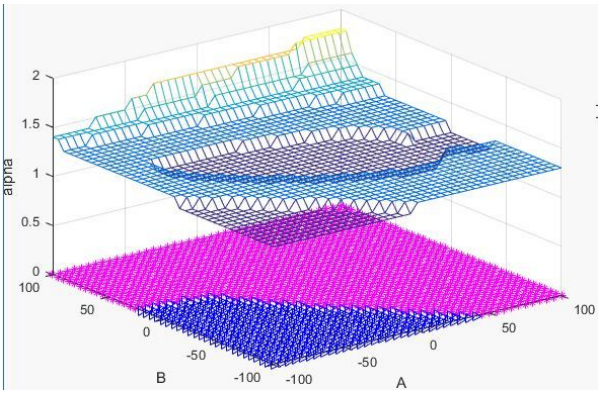



Figure 24: Bist local alpha values @ L=65

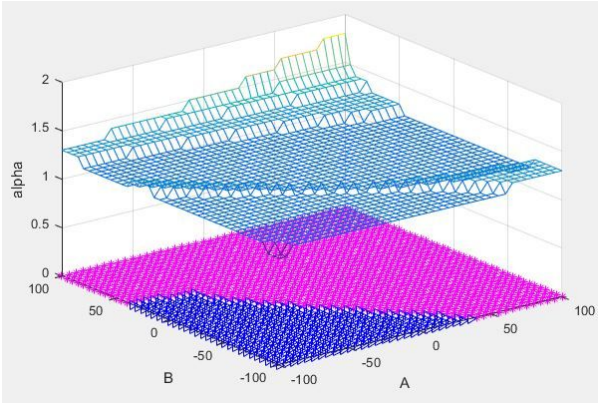



Figure 25: Bist local alpha values @ L=70

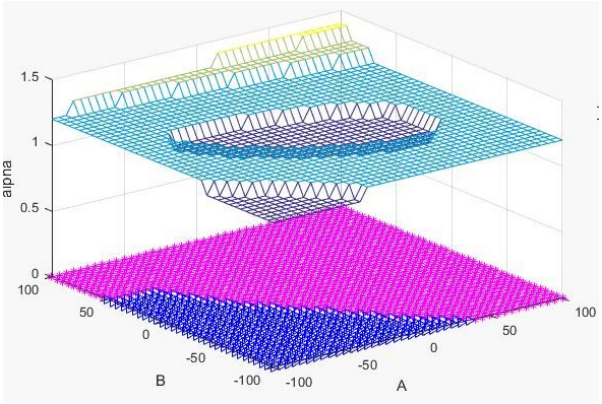



Figure 26: Bist local alpha values @ L=75

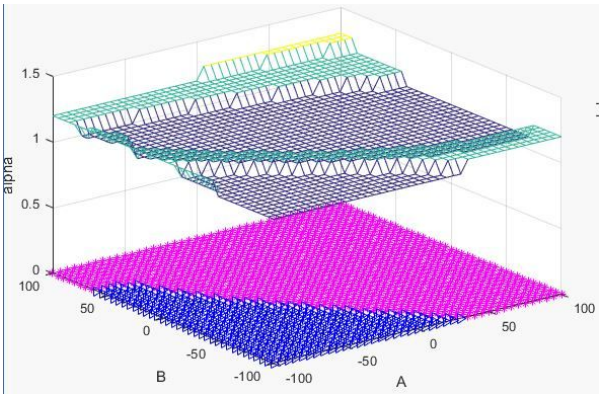



Figure 27: Bist local alpha values @ L=80

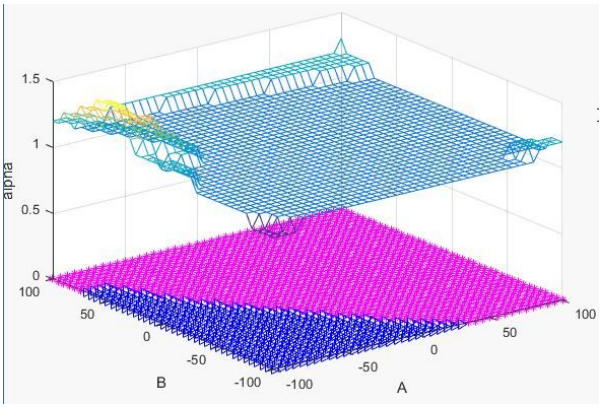



Figure 28: Bist local alpha values @ L=85

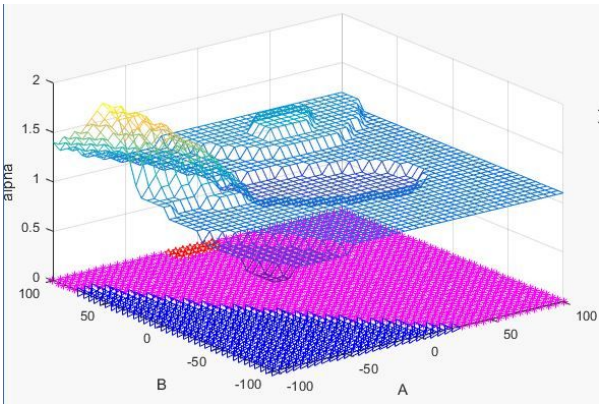



Figure 29: Bist local alpha values @ L=90

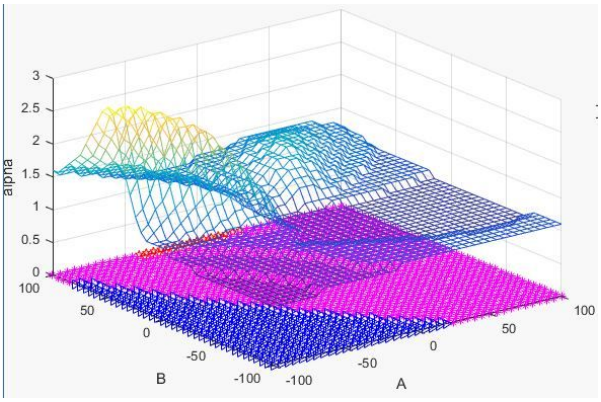



Figure 30: Bist local alpha values @ L=95

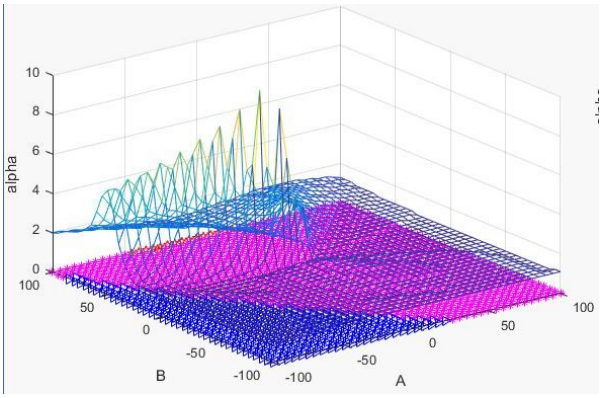



Figure 31: Bist local alpha values @ L=100

### Akyuz Alpha Map 3D Graphs

The local alpha values for Akyuz iTMO are shown in figures 32 to 52. From the Akyuz alpha map 3D graphs, we observed that at lumiance level 0 to 45, the local alpha values remain at 1.

No particular changes in local alpha values are observed at color region boundary between magenta and blue.

Additionally, as the luminance level increases from 0 to 35, the value of alphas remain at 1. And as luminance level further increases from 40 to 100, the local alpha values at the lowest end of A value and the highest end of B value increases.

Under the time limit of the project, the observations above are not further investigated.

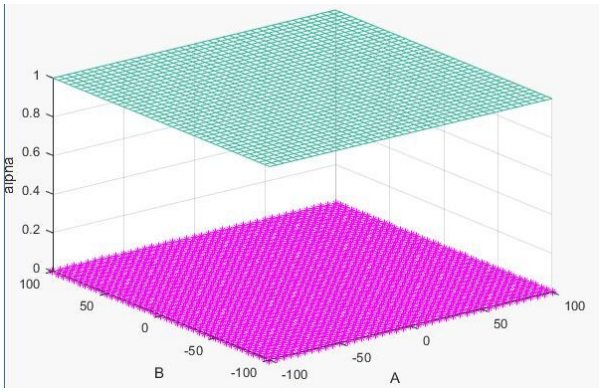



Figure 32: Akyuz local alpha values @ L=0

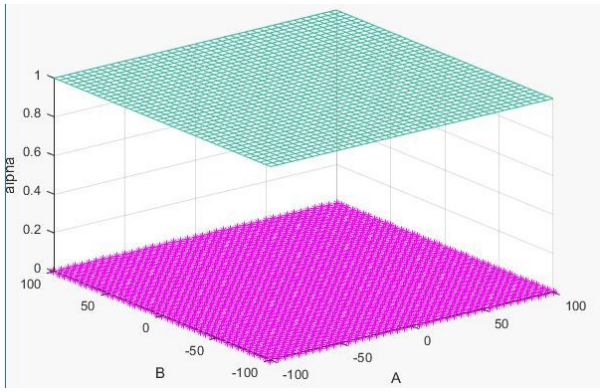



Figure 33: Akyuz local alpha values @ L=5

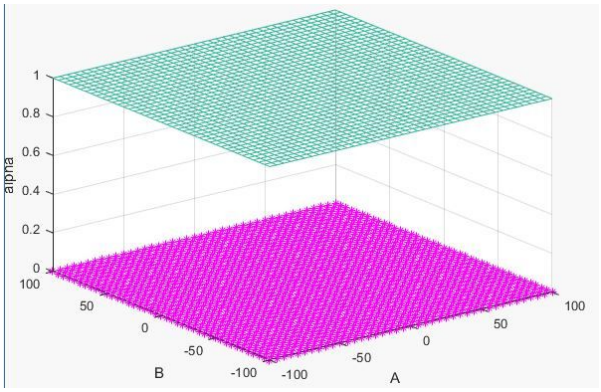



Figure 34: Akyuz local alpha values @ L=10

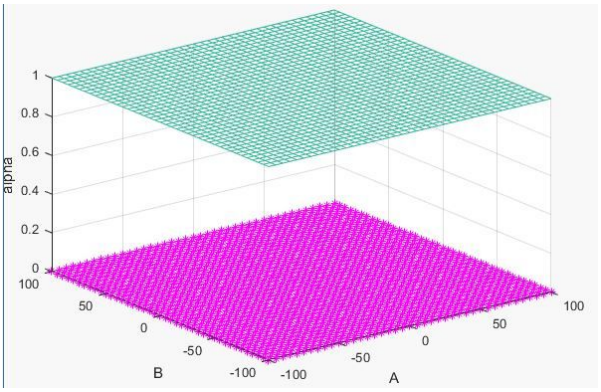



Figure 35: Akyuz local alpha values @ L=15

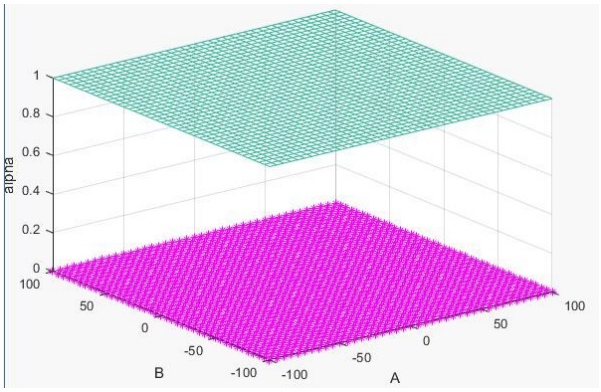



Figure 36: Akyuz local alpha values @ L=20

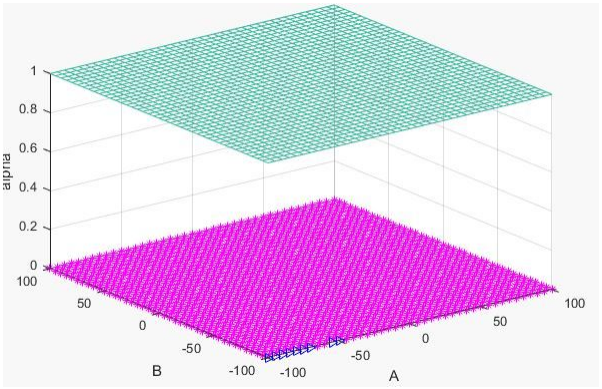



Figure 37: Akyuz local alpha values @ L=25

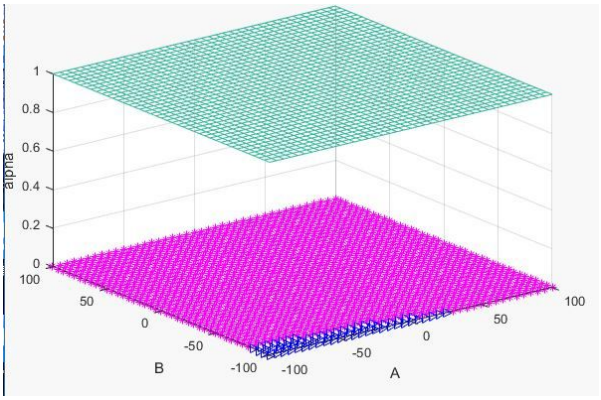



Figure 38: Akyuz local alpha values @ L=30

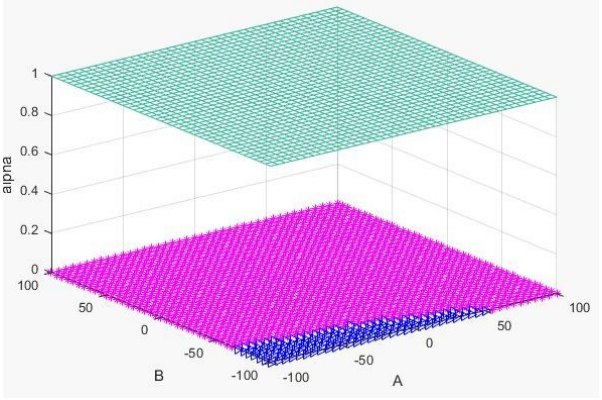



Figure 39: Akyuz local alpha values @ L=35

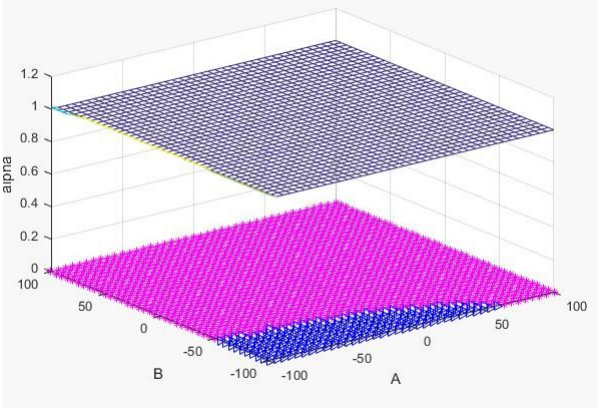



Figure 40: Akyuz local alpha values @ L=40

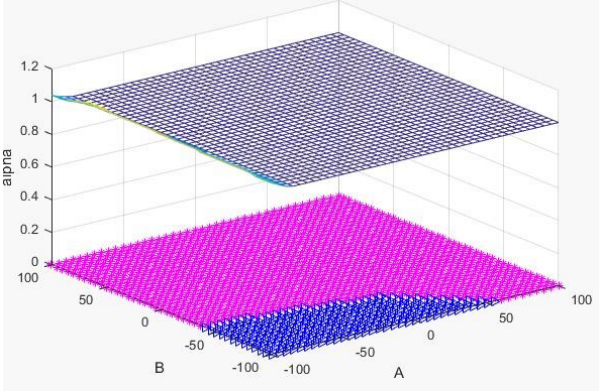



Figure 41: Akyuz local alpha values @ L=45

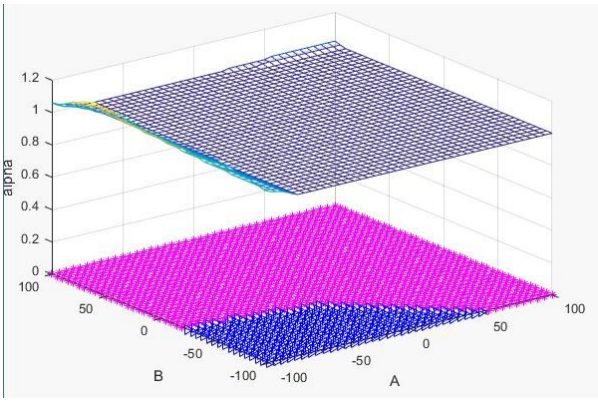



Figure 42: Akyuz local alpha values @ L=50

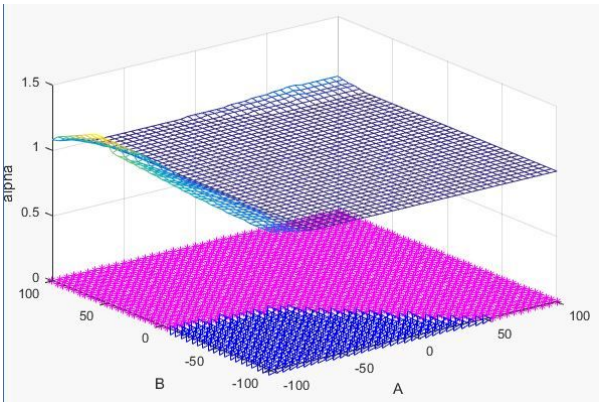



Figure 43: Akyuz local alpha values @ L=55

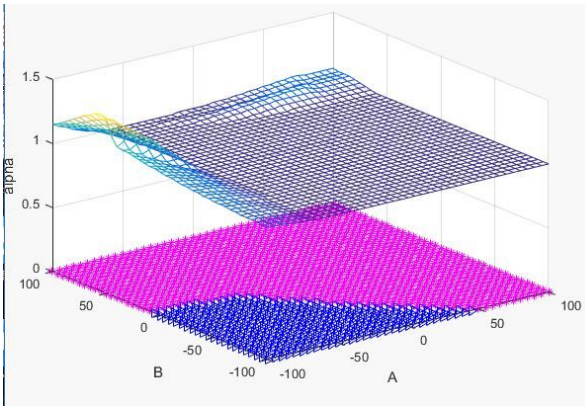



Figure 44: Akyuz local alpha values @ L=60

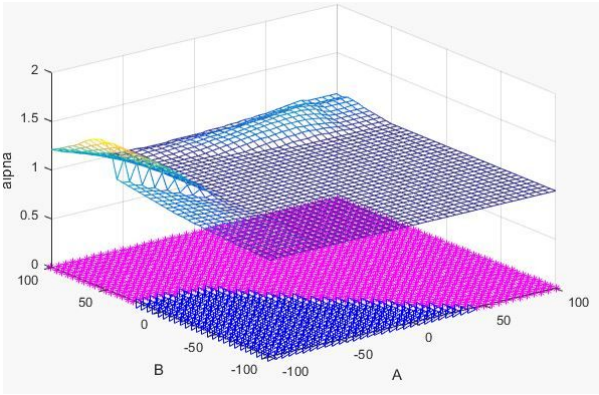



Figure 45: Akyuz local alpha values @ L=65

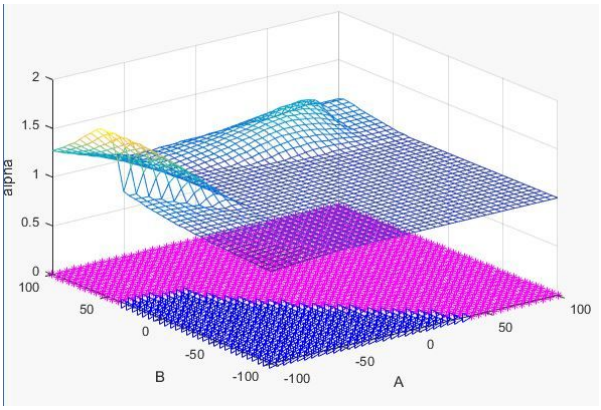



Figure 46: Akyuz local alpha values @ L=70

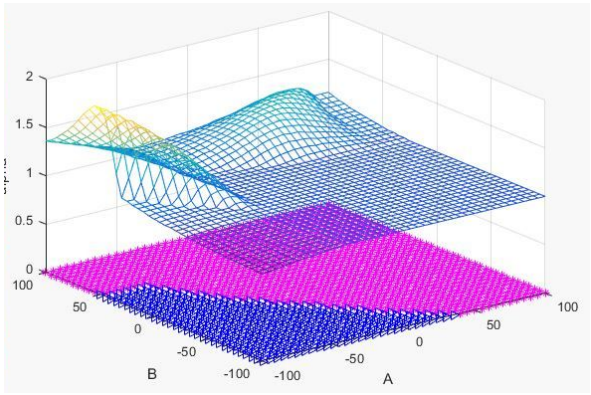



Figure 47: Akyuz local alpha values @ L=75

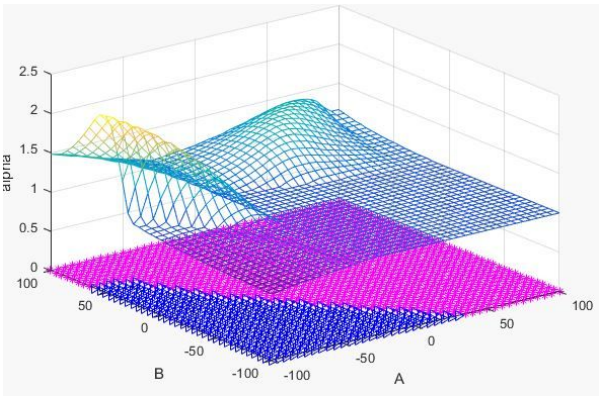



Figure 48: Akyuz local alpha values @ L=80

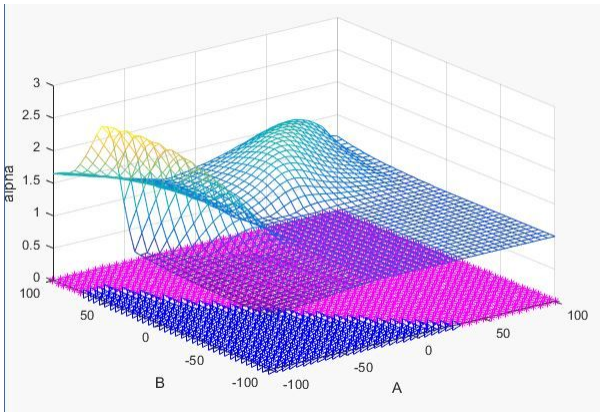



Figure 49: Akyuz local alpha values @ L=85

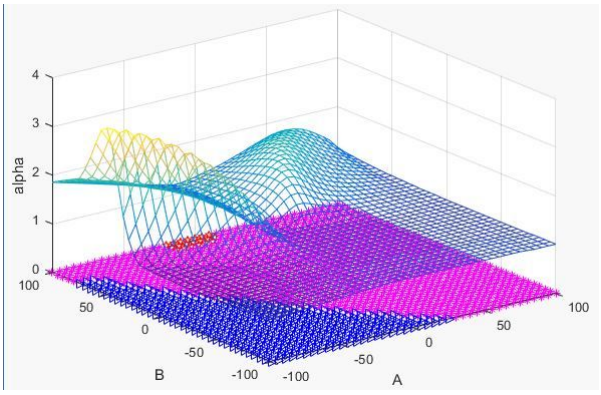



Figure 50: Akyuz local alpha values @ L=90

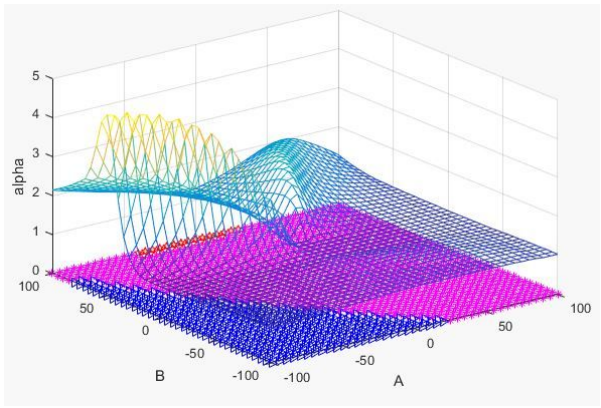



Figure 51: Akyuz local alpha values @ L=95

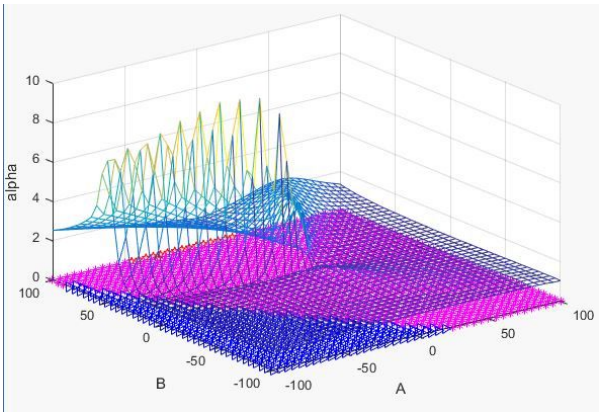



Figure 52: Akyuz local alpha values @ L=100

## Alpha Map 2D Graphs

The 2D graphs at different A & B values displays the trend of of alpha with the change of luminance level. In the 2D graphs, x-axis is the luminance level and y-axis is the local alpha value. The 2D graphs are organized in a grid, the x-axis of the grid represents the A value and y-axis represents the B value.

### Bist Alpha Map 2D Graphs

The local alpha values against luminance level for Bist iTMO are shown in figures 52 and 54. From the Bist alpha map 2D graphs, we observed that the local alpha value generally decreases with the increase in luminance level. The local alpha value decreases most drastically when luminance increases from 0 to 20.

Additionally, it is observed that the shape of curve on the 2D graphs remains similar when moving across different B values at the same A value. However, same cannot be observed for the shape of curve on the 2D graphs when moving across different A values at the same B value. We suggest that, as future work, a mathematical expression can be used to represent the curve across different B values at the same A value with scaling factor to generalized the curve.

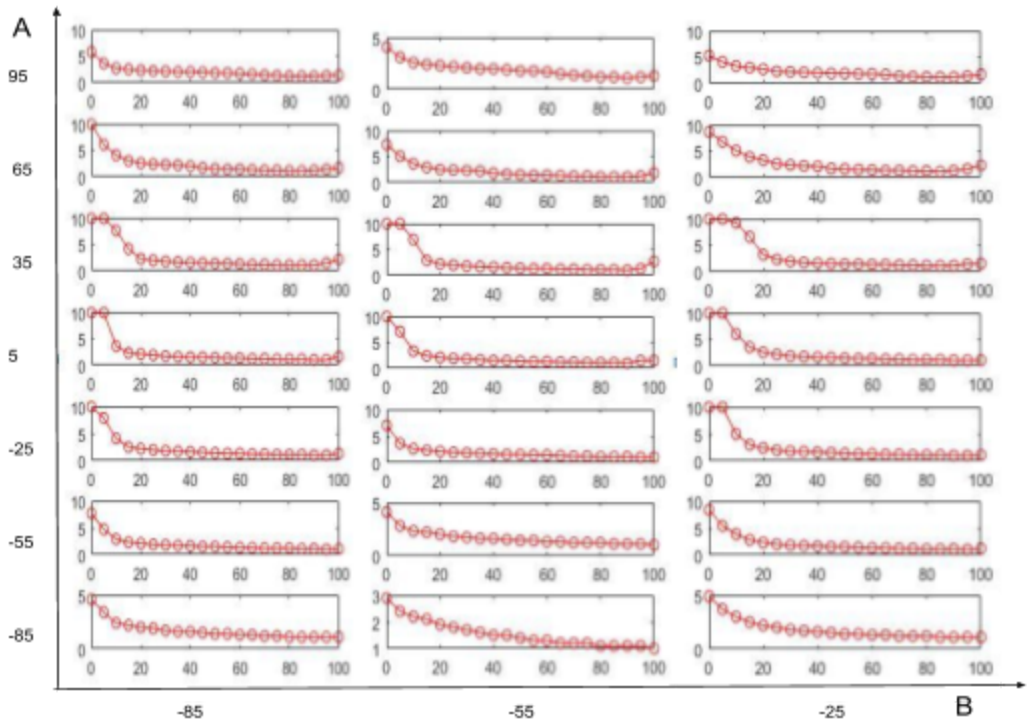



Figure 53: Bist local alpha values against luminance @ A=[-85 to 95] & B=[ -85 to -25]

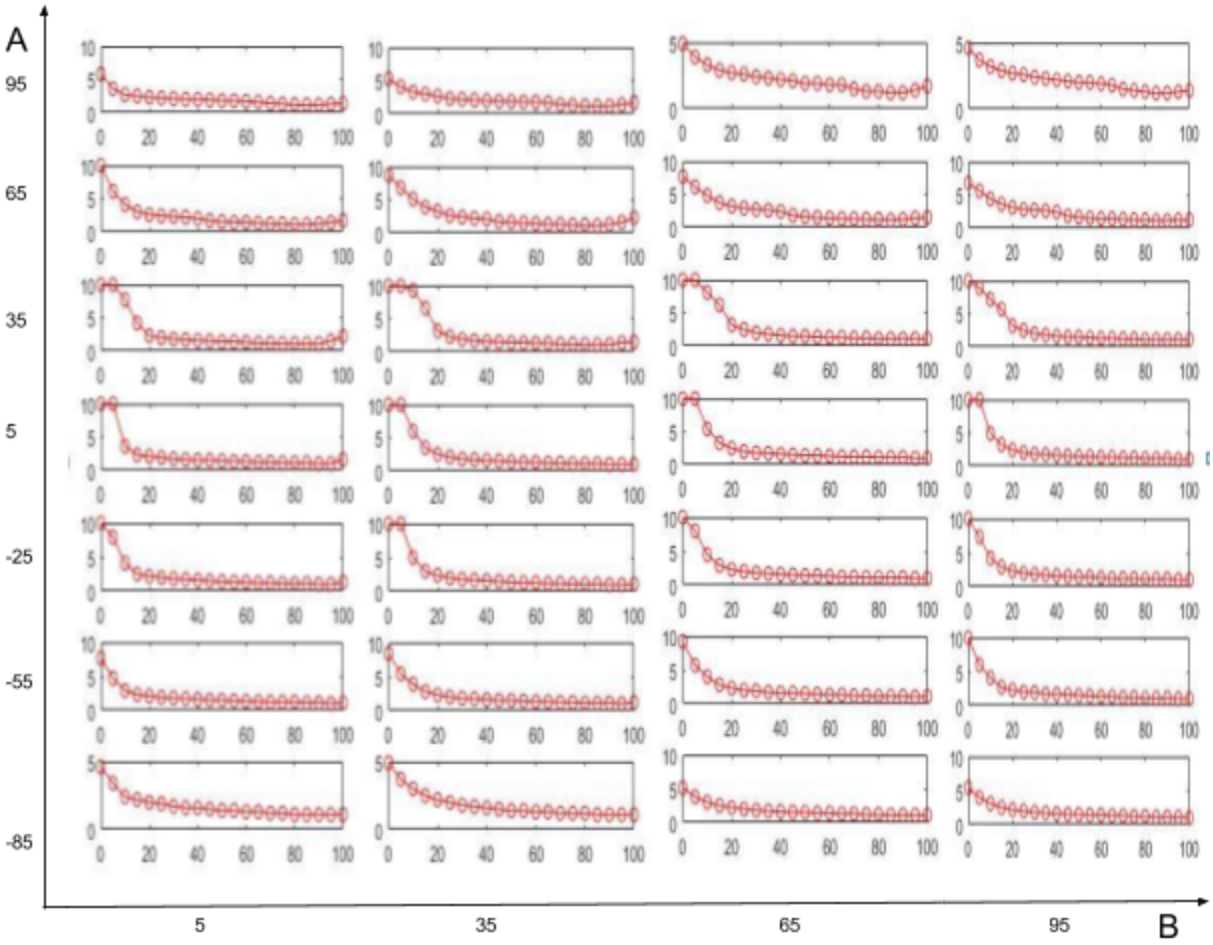



Figure 54: Bist local alpha values against luminance @ A=[-85 to 95] & B=[ 5 to 95]

### Akyuz Alpha Map 2D Graphs

The local alpha values against luminance level for Akyuz iTMO are shown in figures 55 and 56. From the Akyuz alpha map 2D graphs, we observed that majority of local alpha values remain around 1 as luminance increases from 0 to about 70. As luminance level increases further from 80 to 100, majority of the local alpha values display a rapid increase while some show a decrease and increase back to around 1. In terms on curve shapes, no observable patterns are found.

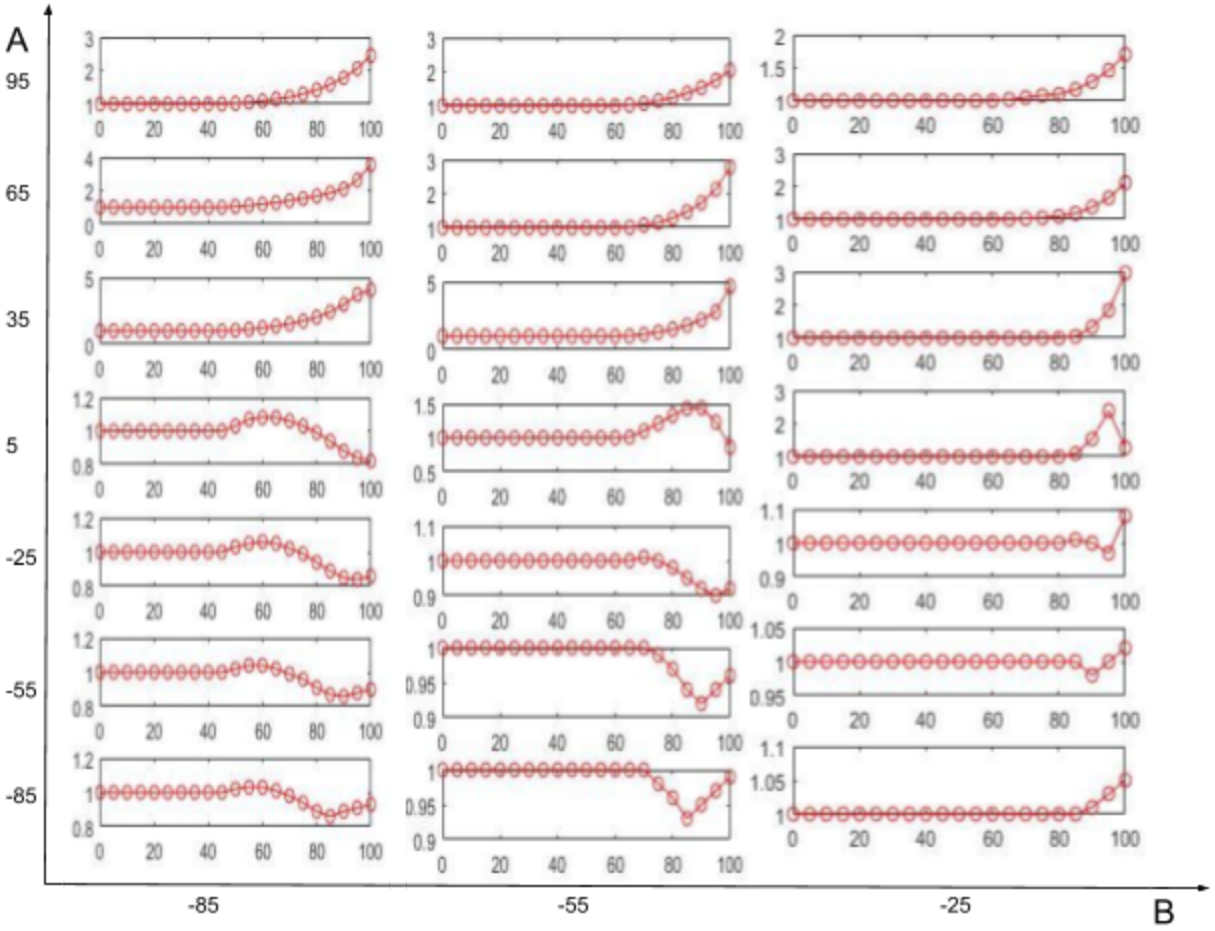



Figure 55: Akyuz local alpha values against luminance
@ A=[-85 to 95] & B=[ -85 to -25]

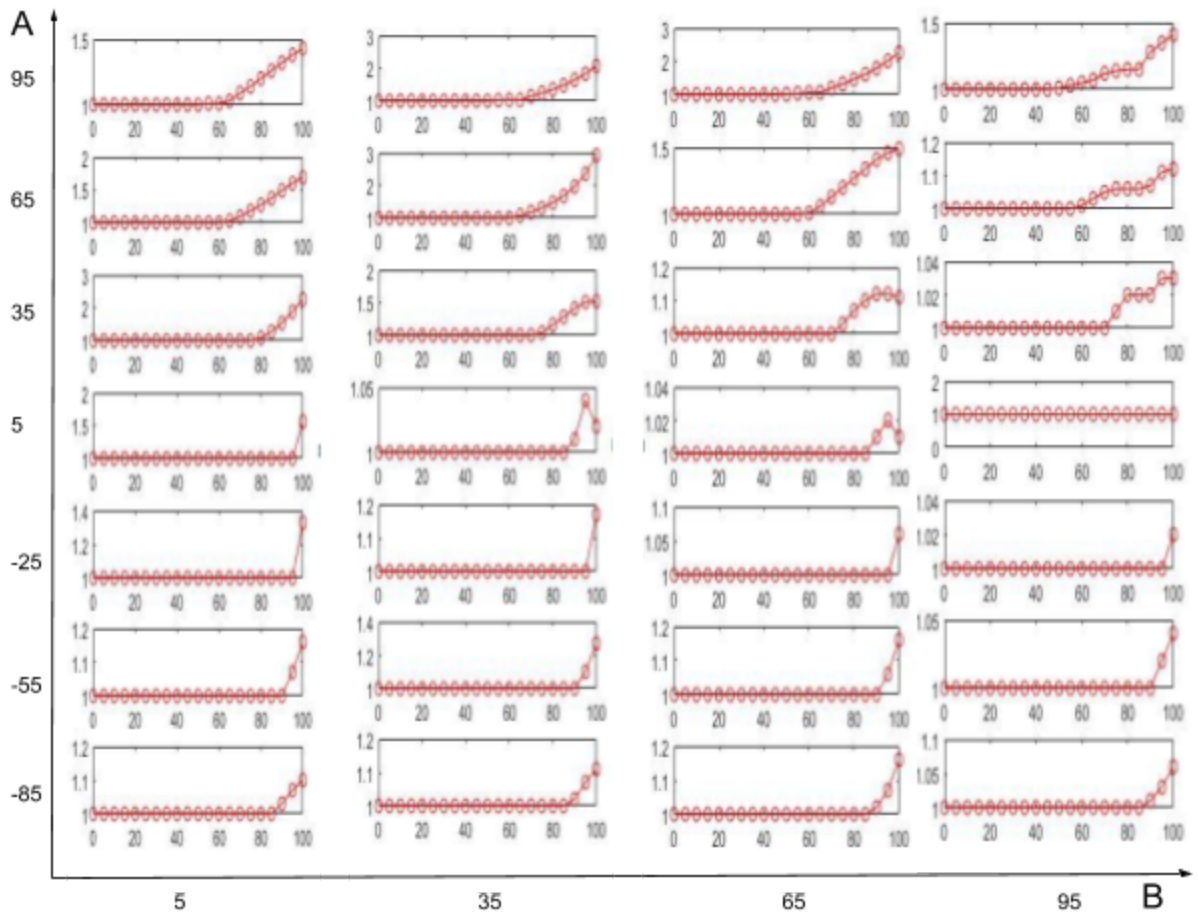



Figure 56: Akyuz local alpha values against luminance @ A=[-85 to 95] & B=[ 5 to 95]

## Alpha Map Histograms

The alpha maps that contains local alpha values for Bist and Akyuz iTMO are displayed in histograms to display the frequency of alpha values in the alpha map.

### Bist Alpha Map Histogram

Figure 57 shows the histogram of alpha values in the alpha map for Bist iTMO with step=5. The alpha values range from 0 to 10. From the figure, we can obseve that most of the alpha values concentrate in the range of 1 to 1.3. In addition, the alpha value 10 shows a high occurrence at about 1000. We suspect that this is because local alpha values beyond have taken 10 as the alpha value as the find alpha map algorithm does not proceed pass the value of 10.

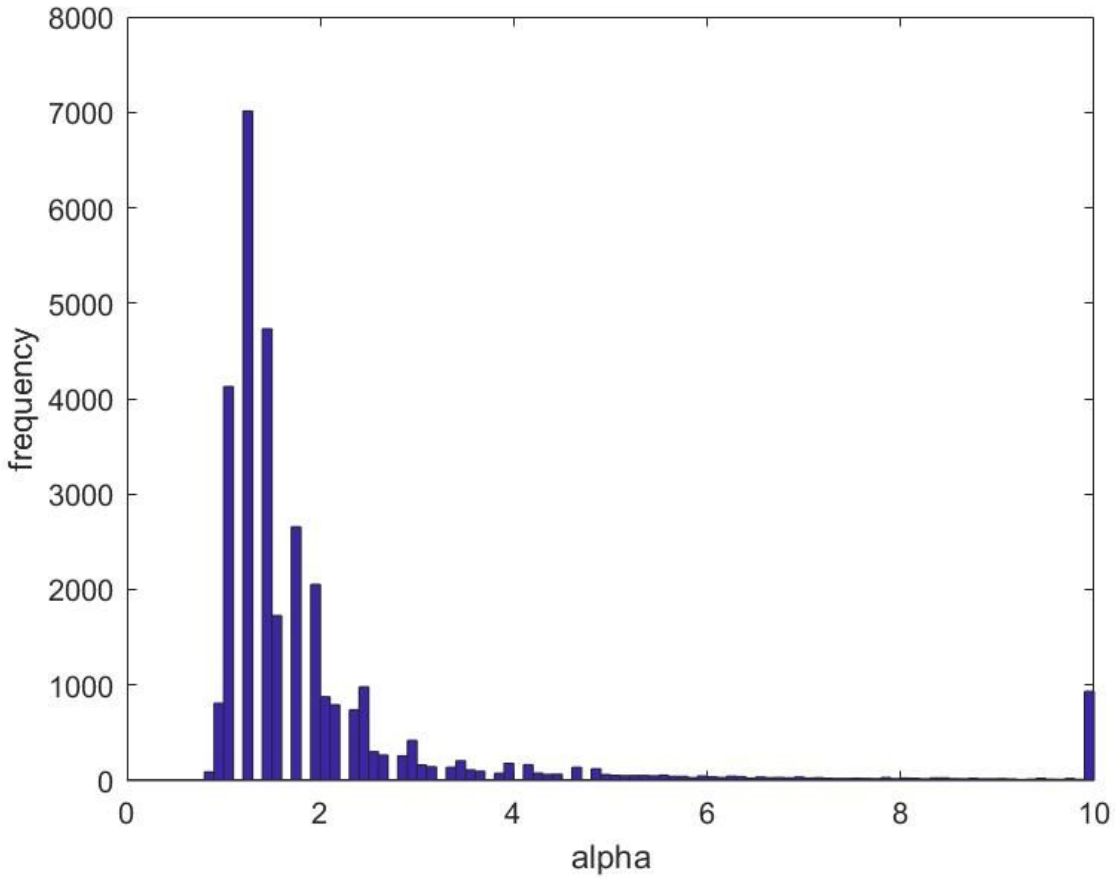



Figure 57: Histogram of alpha values in alpha map for Bist iTMO with step = 5

### Akyuz Alpha Map Histogram

Histogram is also created for the alpha map for Akyuz iTMO. Figure 58 shows the histogram of alpha values in alpha map for Akyuz iTMO with step=5.

From the histogram, we can observe that local alpha value=1 has the highest occurrence in the alpha map. This is consistent with what we observed in 3D & 2D graphs and means that the colour correction adjust with this alpha map would not alter majority of the A and B values of the HDR image.

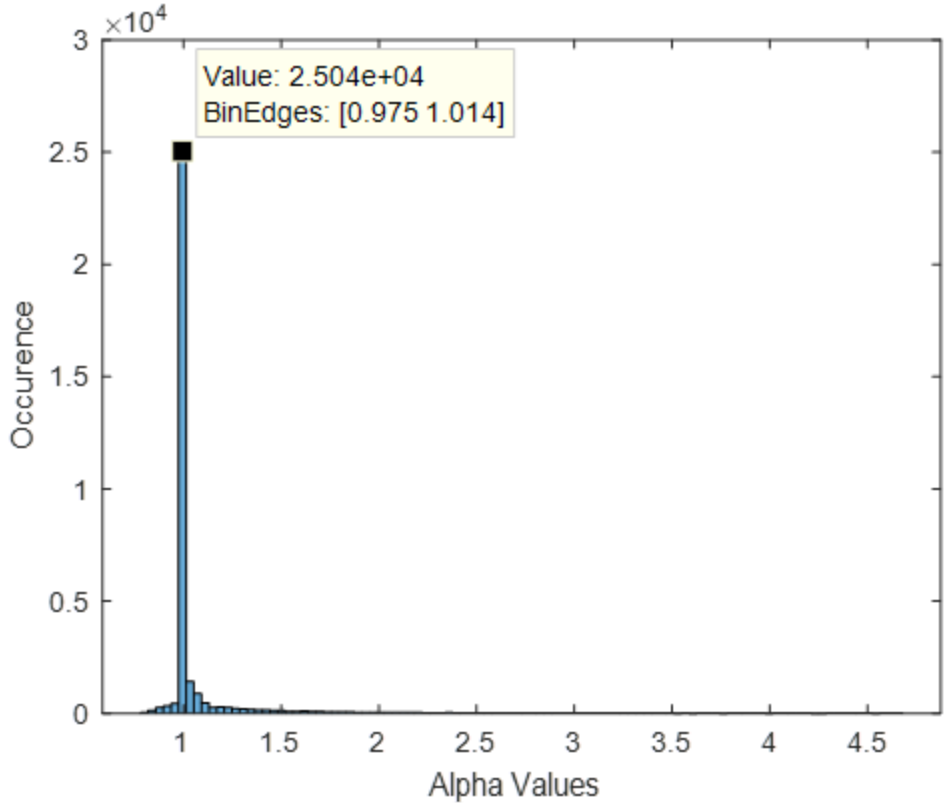



Figure 58: Histogram of alpha values in alpha map for Akyuz iTMO with step = 5

# Colour Correction Results

The application of colour correction method on HDR images from Akyuz and Bist iTMO results in an improvement of colour resemblance of the HDR images. The results are presented in detail in this section.

## Colour Correction Results for Bist iTMO HDR Images

Different test images receive different level of improvement after the colour correction adjustment. One test images is used in this section to show some observable improvements. Images of the test frame at different stages are shown in these figures:

- Figure 59 shows the input SDR image of the test frame.
- Figure 60 shows the output HDR image of Bist iTMO method.
- Figure 61 shows the colour-corrected HDR image using global alpha of 2.1.
- Figure 62 shows the colour-corrected HDR image using local alpha map of step 20.
- Figure 63 shows the colour-corrected HDR image using local alpha map of step 5.

The output HDR images of Bist iTMO loses detail textures in multiple areas such as at the tree line and the surface of the wood piece, the bright part of the mountain range. The contrast of the HDR image is higher than that of the original SDR image.

The color corrected HDR image with global alpha of 2.1 shows a better chroma performance compared to the Bist iTMO output HDR image. The colors shown on the tree and on the lake surface are closer to that of the original SDR images. However, yellow patches appears at regions that are bright in the original SDR image. We suspect the application of a global alpha value improves majority of the HDR image but results in degradation in minority of image.

The application of local alpha maps further improves the colour corrections results. The overall colour of the colour-corrected HDR image resemble more to the original SDR image. The detail texture of the reflection, mountain range and wood surface improves compared to the HDR image adjusted with a single global alpha value.

As HDR image is colour corrected with local alpha map of step 5 instead of step 20, improvements in terms of observable artefacts can be seen. The blue patches appears on the left mountain range and on the reflection of the left side on the lake are absent in the step 5 colour corrected image. In addition, the green patches on the reflection on right side of the lake are removed.

We suspect that the appearance of blue and green patches are caused by the adjusted LAB values exceeding the boundary when converted from LAB colour space to RGB colour space. And the application of local alpha map of step 5 mitigate this issue and remove the patches.

Figure 59: Input SDR image

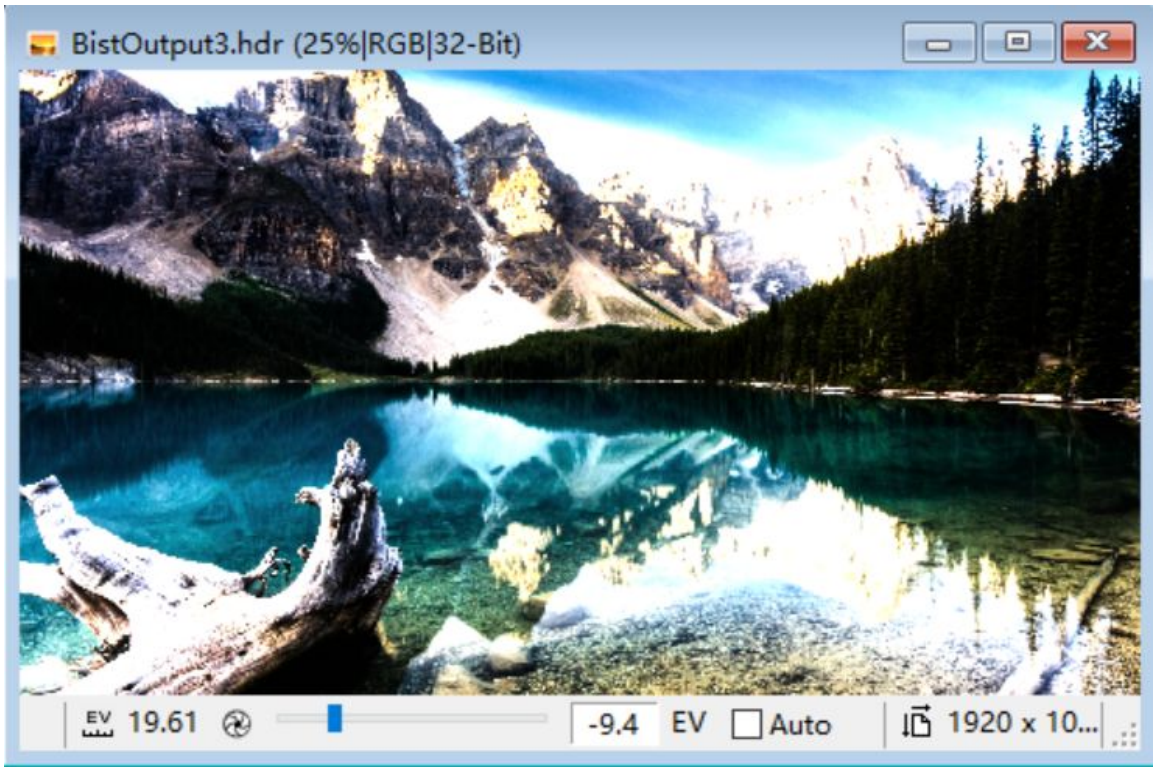



Figure 60: Bist iTMO output HDR image

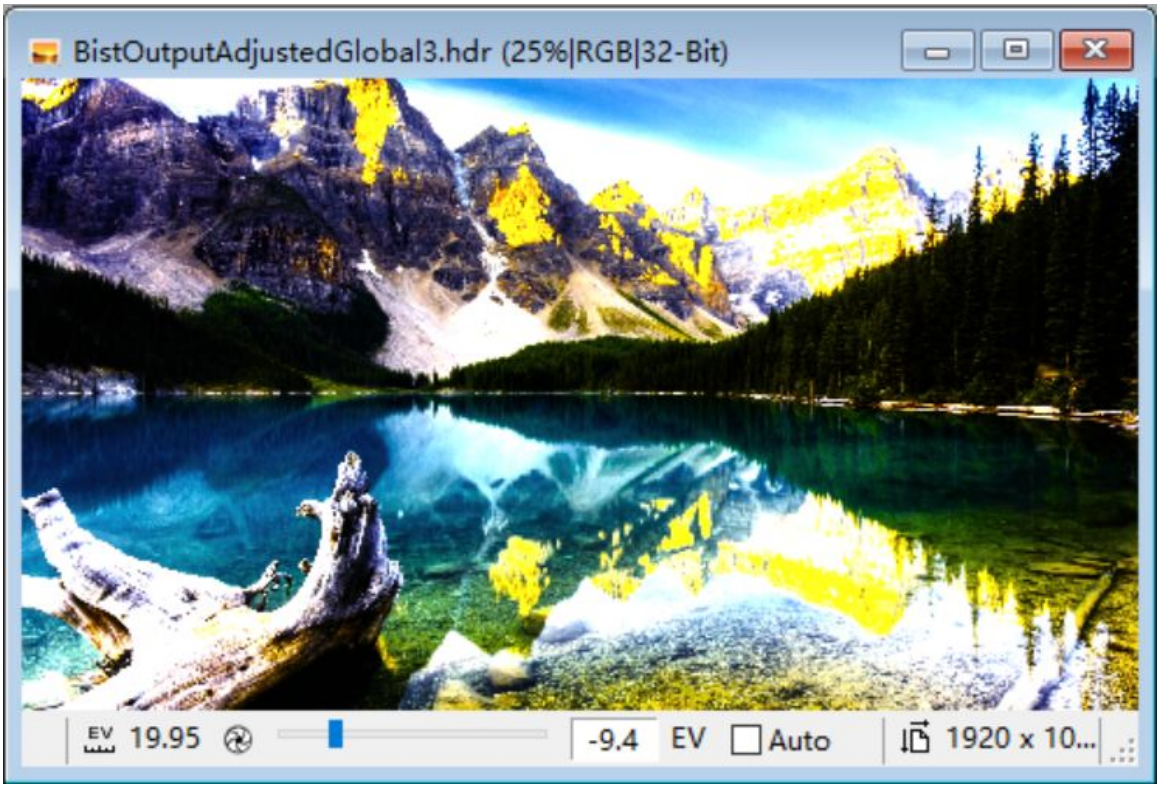



Figure 61. Color-corrected HDR Image with global alpha of 2.1

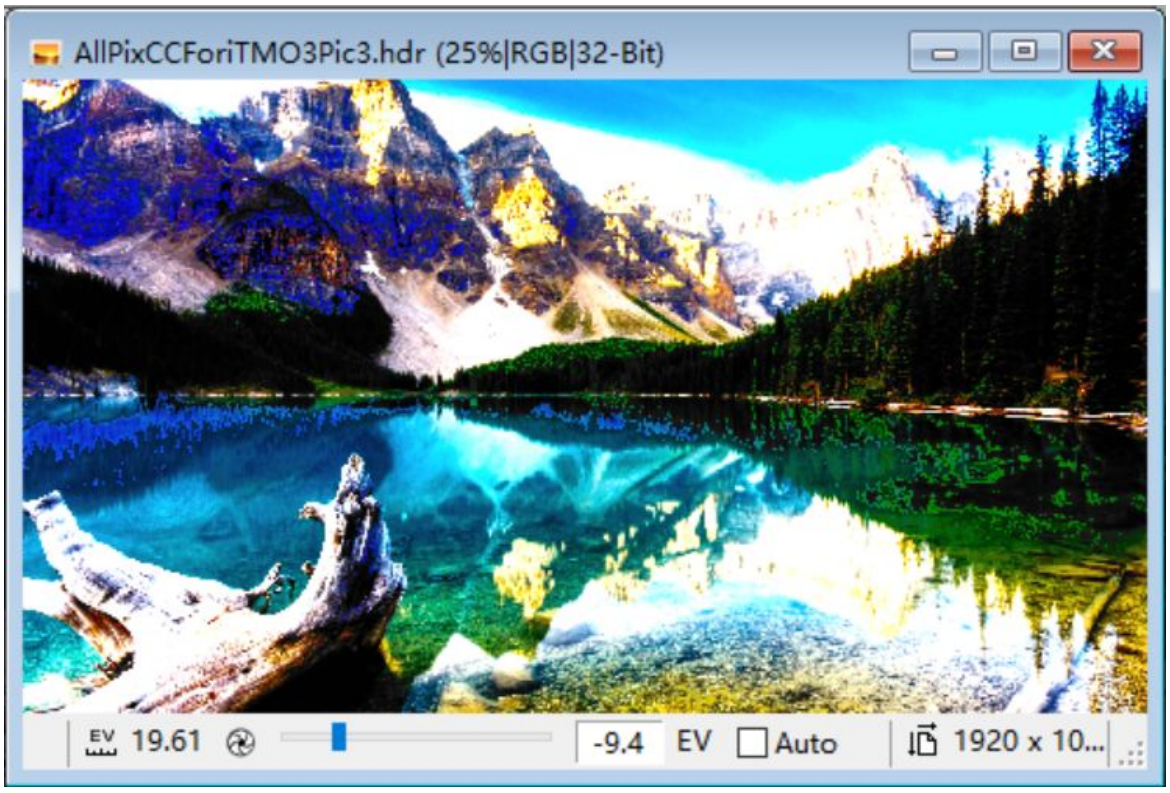



Figure 62: Colour-corrected HDR images with local alpha map @ step=20

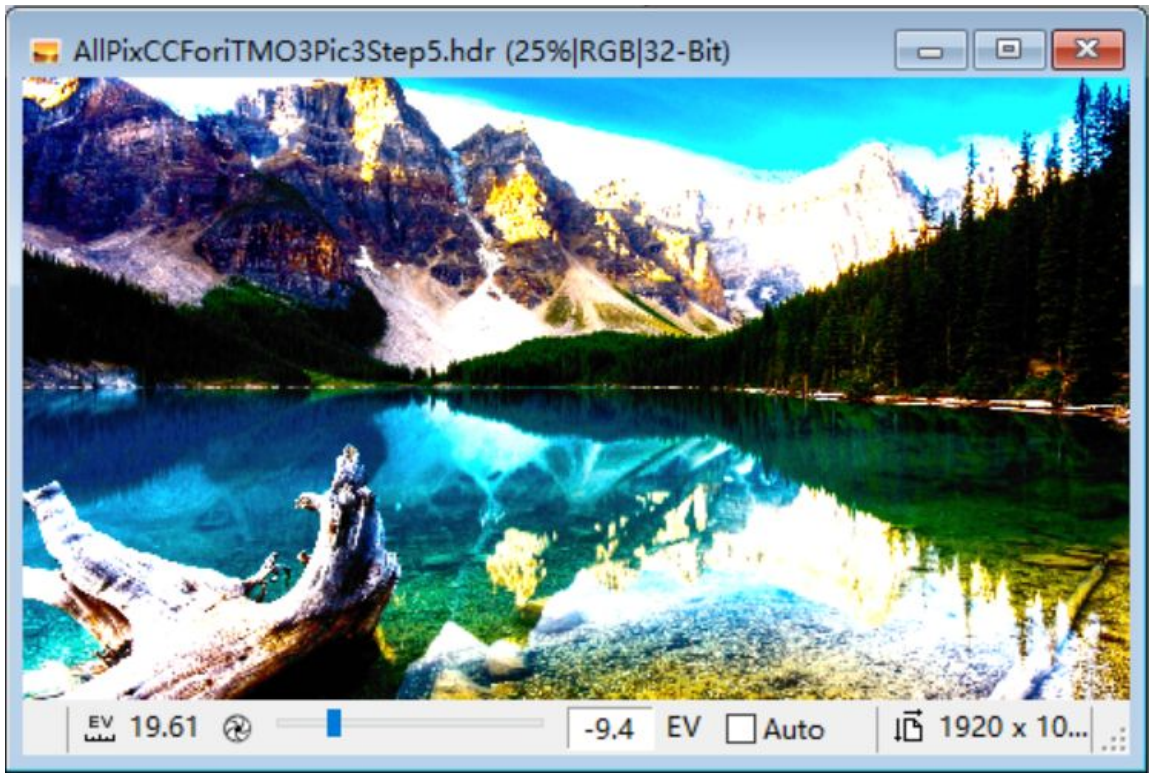



Figure 63: Colour-corrected HDR images with local alpha map @ step=5

### DeltaE Values

Other than the observable artefacts that represent the improvements from the colour correction adjustment, deltaE values are used to determine the improvement. Table 2 shows the deltaE values between the original SDR image and HDR images that are processed differently after the Bist iTMO.

From the deltaE values in table 2, we can observe that not all the deltaE on “global alpha” column is smaller than those on the “HDR Direct” column. This suggests that the application of colour correction with global alpha value on Bist iTMO resulting HDR image might not always result in an improvement of colour resemblance.

Nonetheless, the deltaE values in both the “local alpha” columns are less than those in the “HDR direct” and “global alpha” column. This suggests that the application of colour correction with local alpha values result in an improvement of colour resemblance.

It is expected that the deltaE values of the HDR images colour-corrected with local alpha map of smaller step would be less that that of the ones of larger step. However, as seen in table 2, the deltaE values in “local alpha step=20” column is generally less than those in the “local alpha step=5” column. We are puzzled with this finding, but have yet to found the cause of this result.

| Test Frames | HDR Direct | Global Alpha | Local Alpha step=20 | Local Alpha step=5 |
|---|---|---|---|---|
| 1- Chair | 47247.69 | 46720.58 | 42681.57 | 43453.15 |
| 2 - Car | 32117.87 | 37694.96 | 30673.34 | 30764.62 |
| 3 - Lake | 39114.81 | 41045.67 | 36060.55 | 36499.56 |
| 4 - Family | 32731.43 | 23542.00 | 31650.06 | 31707.26 |
| 5 - Garden | 35286.77 | 44345.71 | 32724.77 | 32908.77 |
| 6 - Home | 40990.19 | 41572.20 | 39541.89 | 39553.83 |
| 7 - Restaurant | 25657.14 | 38512.54 | 23343.16 | 24060.28 |

Table 2: DeltaE values between original SDR image and HDR images that are processed differently after Bist iTMO.

### Issue with Colour Correction Adjustment with Bist Local Alpha Map

With one test frame, we noticed the appearance of dominating red regions as show in figure 64. This is caused by the high occurrence of local alpha value=10 as shown in figure 65.

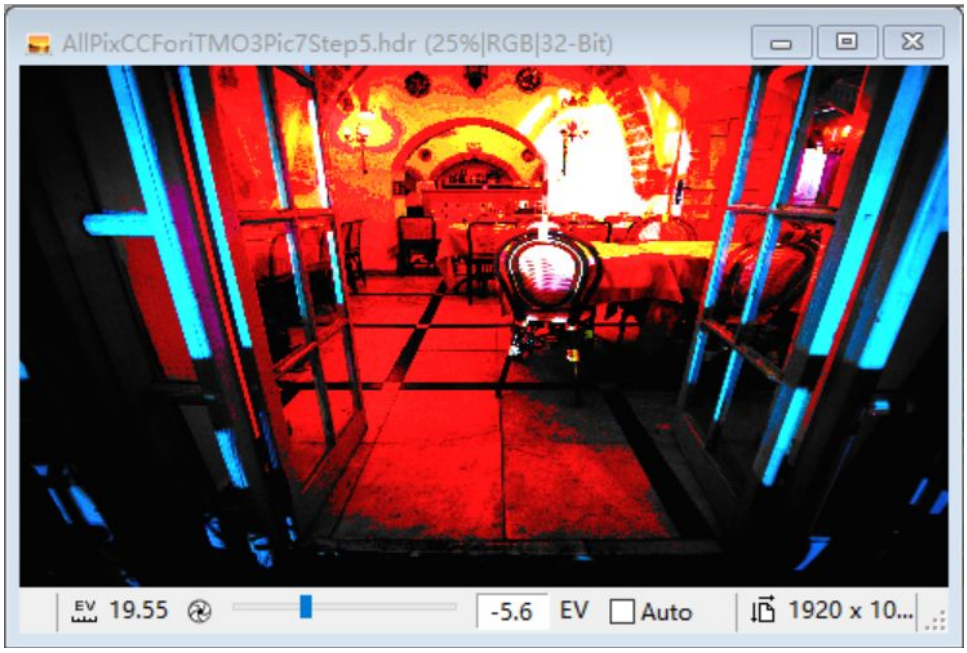



Figure 64: Appearance of dominating red colour region in colour corrected HDR image

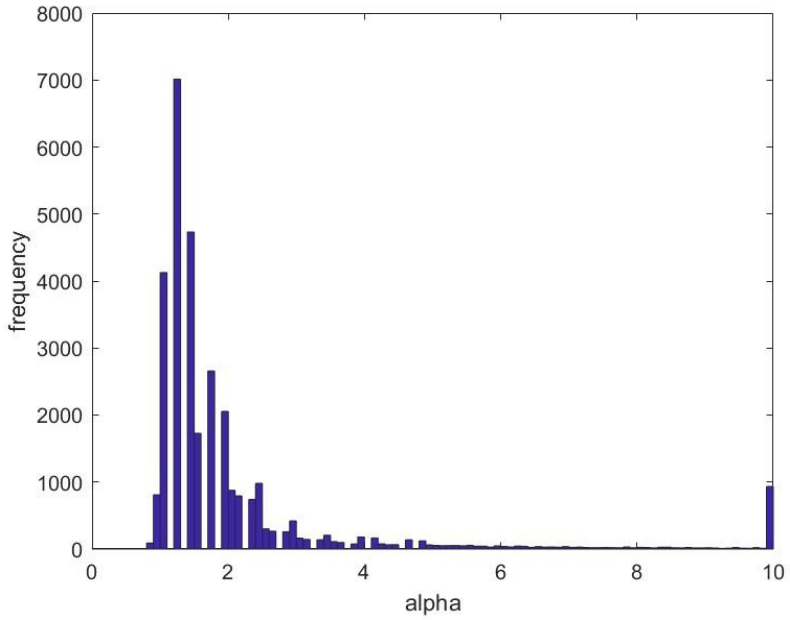



Figure 65: Histogram of alpha values in alpha map for Bist iTMO with step = 5

To mitigate the issue, for the local alpha values that are 10, 1 is used instead for the colour adjustment. The images in figure 66 is produced. We can observe the disappearance of the red regions comparing to the image in figure 64.


Figure 66: Removal of dominating red colour region in colour corrected HDR image

### Range of A & B Values

In order to further visualize the the effect of iTMO and colour correction adjustment to the colour of the image, graphs of the AB plane of the LAB colour space are created to visualize the A & B value changes along the process.

The graph in figure 67 shows the AB plane of the LAB colour space. In the graph, there are three regions.

1. Blue region that is determined by the maximum value and minimum value of A and B of the original SDR images.
2. Red region that is determined by the maximum value and minimum value of A and B of the HDR images produced by the Bist iTMO
3. Green region that is determined by the maximum value and minimum value of A and B of the HDR images after colour correction adjustment with local alpha values.

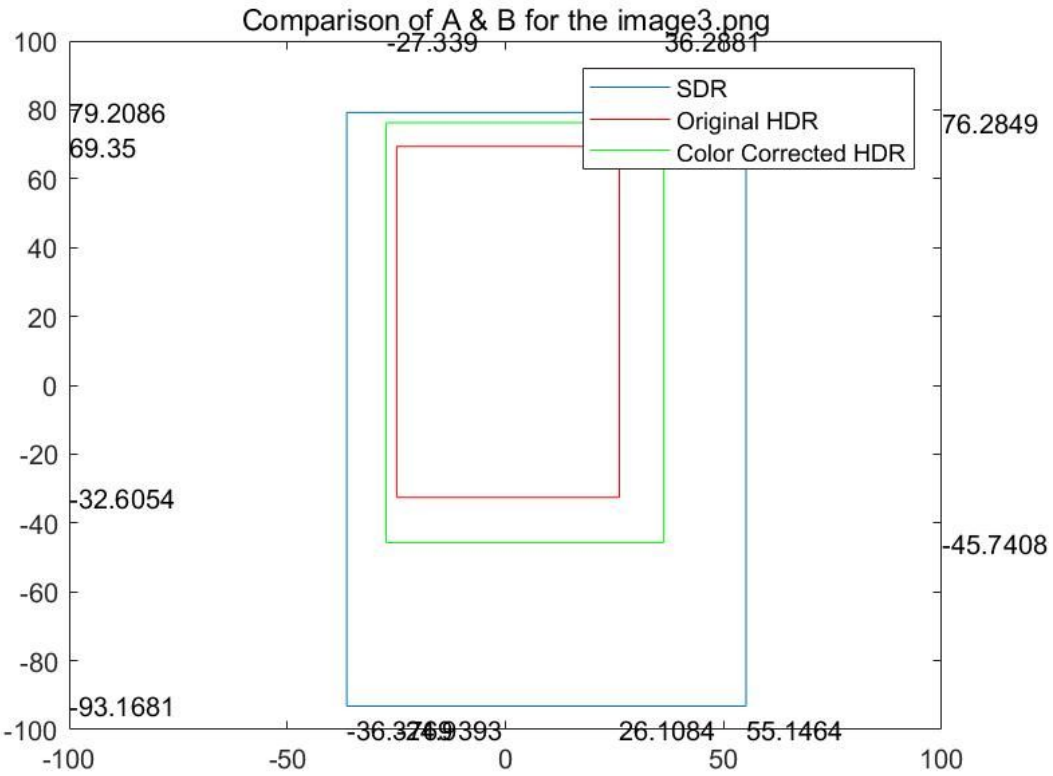



Figure 67: A-B regions of SDR, HDR, and adjusted HDR images of test frame 3 on A-B plane of LAB colour space

From the graph, we observed that after an SDR image (blue region) is processed by an iTMO producing the HDR image (red region), the magnitude of A & B values decrease and occupy a smaller region.

With the color correction adjustment, the HDR image (red region) expands forming the adjusted HDR image (green region). The expansion of red region to green region contributes to the lower of deltaE value of the adjusted HDR image than the HDR image. This is due to the reduction of Euclidean distance of LAB values between the adjusted HDR image and the original SDR image. With the reduction in Euclidean distance, the colour in the colour corrected HDR images perceptually resembles the original SDR images more than that of the non-corrected HDR images.

Additional graphs for other test frames are attached on appendix 14. All of the graphs show the same observation in region changes along the iTMO and colour correction process.

## Colour Correction Results for Akyuz iTMO HDR Images

In this section, the same test image as the Bist iTMO section is used for the discussion of colour correction of image from Akyuz iTMO. Images of the test frame at different stages are shown in these figures:

- Figure 68 shows the input SDR image of the test frame.
- Figure 69 shows the output HDR image of Akyuz iTMO method.
- Figure 70 shows the colour-corrected HDR image using global alpha of 1.
- Figure 71 shows the colour-corrected HDR image using local alpha map of step 20.
- Figure 72 shows the colour-corrected HDR image using local alpha map of step 5.

For the global colour correction, the global alpha value is found to be 1. As a result, the HDR image adjusted does not show any difference when compared to the resulting HDR image of the Akyuz iTMO.

With the colour correction with local alpha values, similar observation is made. The HDR image adjusted does not show any observable differences when compared to the resulting HDR image of the Akyuz iTMO. This is because the majority of local alpha values in both of the alpha maps are 1. A & B values are not alter when alpha value of 1 is applied.

Figure 68: Input SDR image

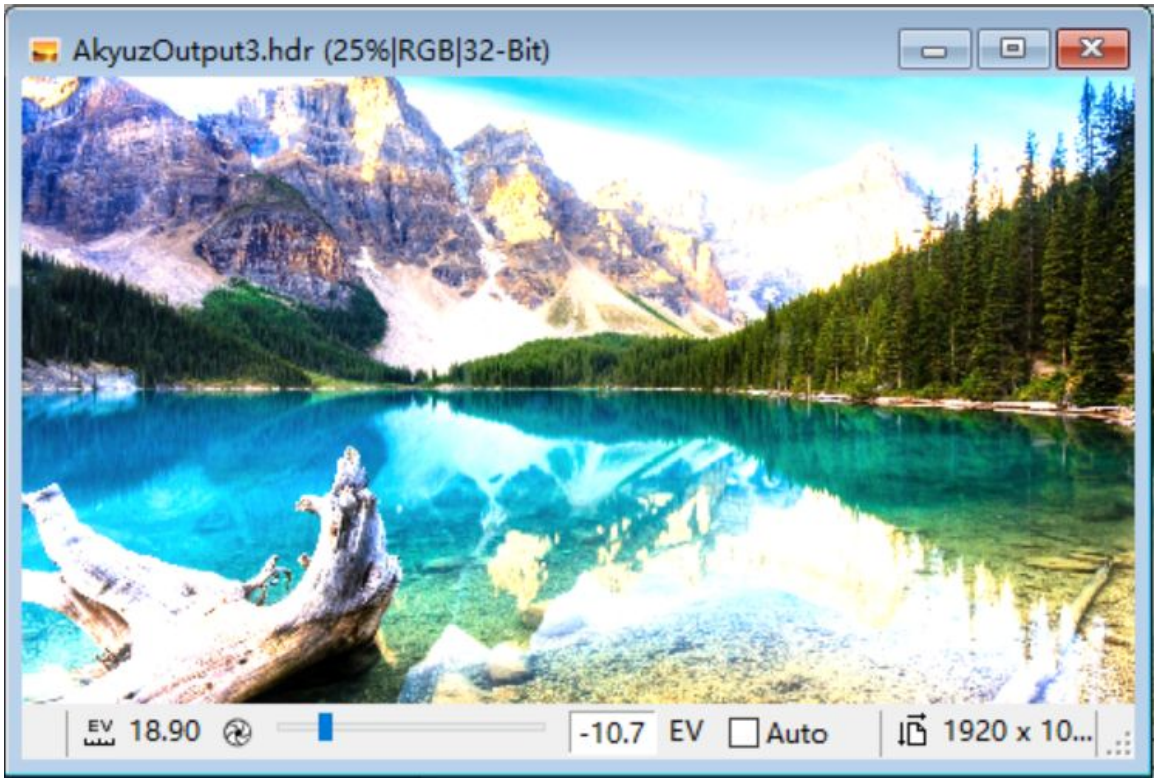



Figure 69: Akyuz iTMO output HDR image.


Figure 70: Color corrected HDR image of Akyuz iTMO with global alpha 1

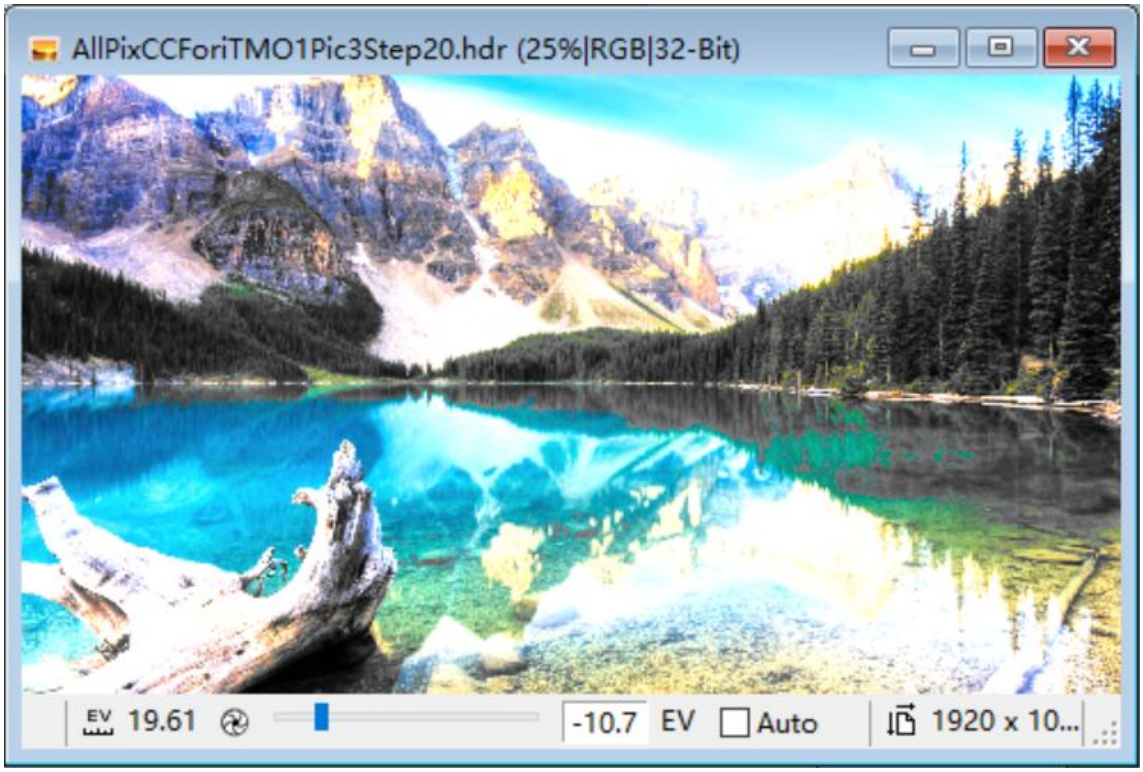



Figure 71: Color-corrected HDR image of Akyuz iTMO with local alpha map @ step=20

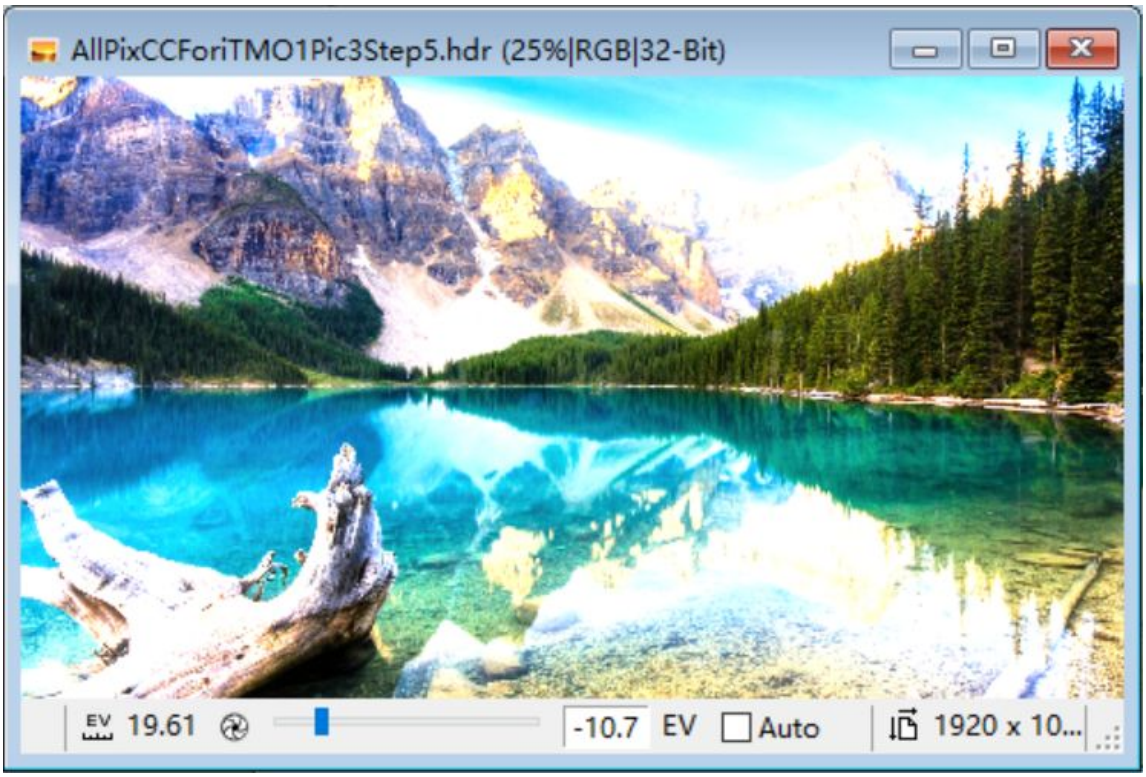



Figure 72: Color-corrected HDR image of Akyuz iTMO with local alpha map @ step=5.

### DeltaE Values

Table 3 shows the deltaE values between the original SDR image and HDR images that are processed differently after the Akyuz iTMO.

From the deltaE values in table 3, we can observe that all the deltaE on “global alpha” column is the same as those on the “HDR Direct” column. This is because the application of global alpha value of 1 does not change the HDR and does not result in an improvement of colour resemblance.

Another finding is that the deltaE values in the “local alpha step=20” column is significantly greater than those in the “HDR direct” and “global alpha” column. This is the same result as the images show above, it has a lot of strange gray patches. But when using smaller local alpha step as 5, the result is similar to the original one, although they do not always show smaller values. This is the reason why the result of local alpha with step=5 is similar to the original HDR output and the global color correction result.

| Test Frames | HDR Direct | Global Alpha | Local Alpha step=20 | Local Alpha step=5 |
|---|---|---|---|---|
| 1- Chair | 262.46 | 262.46 | 3832.68 | 276.83 |
| 2 - Car | 226.93 | 226.93 | 3429.72 | 216.83 |
| 3 - Lake | 201.47 | 201.47 | 36060.55 | 164.52 |
| 4 - Family | 205.08 | 205.08 | 3573.66 | 90.56 |
| 5 - Garden | 179.06 | 179.06 | 1377.43 | 239.66 |
| 6 - Home | 225.85 | 225.85 | 1836.01 | 117.11 |
| 7 - Restaurant | 262.46 | 487.04 | 4278.46 | 537.52 |

Table 3: DeltaE values of HDR images.

### Range of A & B Values

Graphs of the AB plane of the LAB colour space are created to visualize the A & B value changes along the process. The graph in figure 73 shows the AB plane of the LAB colour space. In the graph, there are three regions.

1. Blue region that is determined by the maximum value and minimum value of A and B of the original SDR images.
2. Red region that is determined by the maximum value and minimum value of A and B of the HDR images produced by the Bist iTMO
3. Green region that is determined by the maximum value and minimum value of A and B of the HDR images after colour correction adjustment with local alpha values.

In figure 73, since the chroma components A and B are not changed with global alpha value of 1 or the alpha map with local alpha values close to 1. The three regions overlap each other in the AB plane graph.

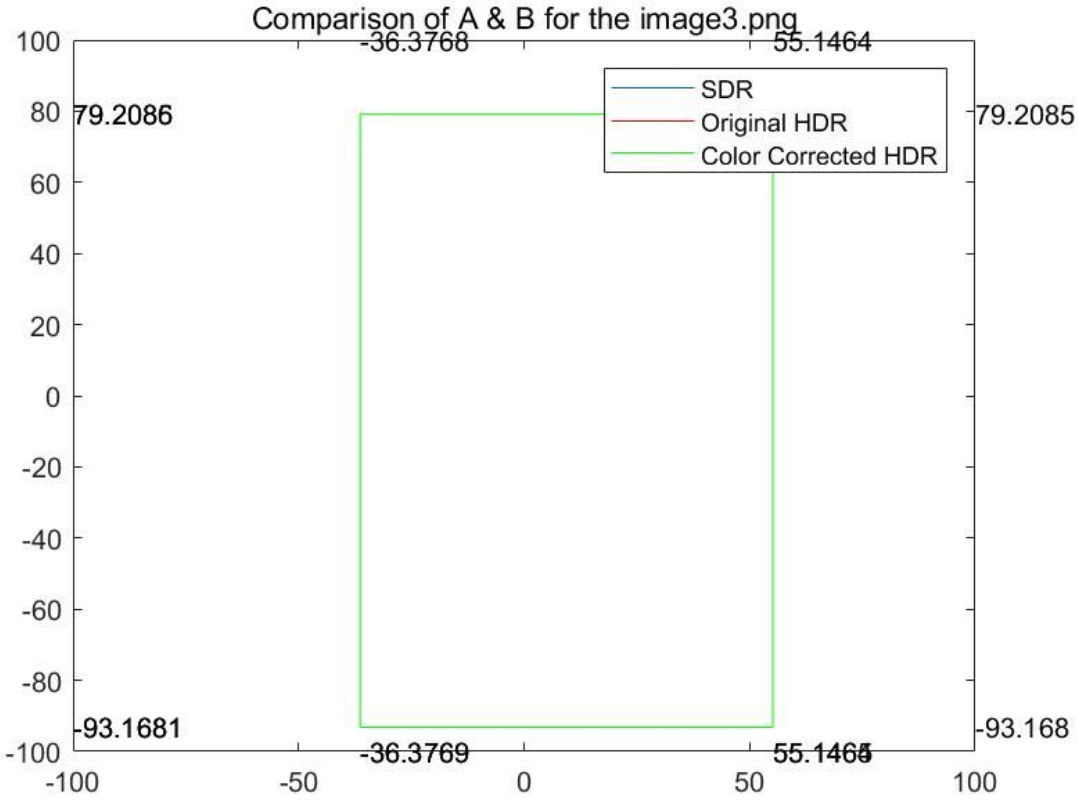



Figure 73: A-B regions of SDR, HDR, and adjusted HDR images of test frame 3 with Akyuz iTMO on A-B plane of LAB colour space

# Conclusion

In this project, our project group implemented two types of colour correction adjustment for the resulting HDR images from 4 iTMO methods.

The global colour correction adjustment applies a global alpha value to all the pixels of the HDR image to minimize the deltaE value between the adjustment result and the original SDR image. The global alpha values are image-specific.

The local colour correction adjustment applies local alpha values to individual pixels of the HDR image to minimize the deltaE value between the adjustment result and the original SDR image. An alpha map is used to contain all the local alpha values for each LAB value. The alpha maps are iTMO-specific.

The global alpha values for an image and the local alpha values for each LAB values are found using trial-and-error approach. The step size of the trial-and-error process affects the improvement results of the colour correction adjustment.

The subjective colour correction results are presented in this report. The color corrected images with global alpha appears with colour closer to the original SDR image. For Bist iTMO HDR images, colour correction with local alpha values results in further improvements in colour resemblance. In terms of objective results, deltaE values suggest colour correction improvements on Bist iTMO resulting HDR images.

The local colour correction adjustment has its limitations. Because it only considers individual pixel values without considering the neighbour pixels. It cannot be readily applied on certain iTMOs including Banterle and Meylan that consider neighbour pixels for the iTMO process. More future effort is needed to generalize the application of the local method on all iTMO results.

# Reference

[1] C. Bist, R. Cozot, G. Madec and X. Ducloux, "Style Aware Tone Expansion for HDR Displays", in Proceedings of the 42nd Graphics Interface Conference (GI '16), Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2016, pp. 57-63.

[2] A. Akyüz, R. Fleming, B. Riecke, E. Reinhard and H. Bülthoff, "Do HDR displays support LDR content?", ACM Transactions on Graphics, vol. 26, no. 3, p. 38, 2007.

[3] F. Banterle, P. Ledda, K. Debattista, A. Chalmers and M. Bloj, "A framework for inverse tone mapping", Visual Computer, vol. 23, no. 7, pp. 467-478, 2007.

[4] M. Laurence, D. Scott and S. Sabine, "Tone Mapping For High Dynamic Range Displays", in SPIE Electronic Imaging: Human Vision and Electronic Imaging XII, SPIE, vol. 6492, pp.1-12,2007.

[5] "Write image to graphics file - MATLAB imwrite - MathWorks United Kingdom", Mathworks.com, 2017. [Online]. Available: http://www.mathworks.com/help/matlab/ref/imwrite.html. [Accessed: 15- Dec- 2017].

[6] "Write Radiance high dynamic range (HDR) image file - MATLAB hdrwrite - MathWorks United Kingdom", Mathworks.com, 2017. [Online]. Available: http://www.mathworks.com/help/images/ref/hdrwrite.html?s_tid=doc_ta. [Accessed: 15- Dec- 2017].

[7] S. Gaurav, Digital Color Imaging Handbook. CRC, 2003, p. 31.

[8] R. Mantiuk, R. Mantiuk, A. Tomaszewska and W. Heidrich, "Color correction for tone mapping", Computer Graphics Forum, vol. 28, no. 2, pp. 193-202, 2009.

[9] T. Pouli, "Color Correction for Tone Reproduction", in Twenty-first Color and Imaging Conference, 2013, pp. 215-220.

[10] A Review of RGB color spaces...from xyY to R'G'B'. (2002). 1st ed. [ebook] Montreal.Canada: The BabelColor Company, pp.3-6.

[11] The C.I.E. colorimetric standards and their use. (1931). Transactions of the Optical Society, 33(3), pp.73-134.

[12] H. Richard Sewall, "Photoelectric Color-Difference Meter", in Winter Meeting of the Optical Society of America, 1948, p. 661.

[13] P. Trezona, "Derivation of the 1964 CIE 10,XYZ colour-matching functions and their applicability in photometry", Color Research & Application, vol. 26, no. 1, pp. 67-75, 2000.

[14] "CIELab Color Space", Docs-hoffmann.de, 2017. [Online]. Available: http://docs-hoffmann.de/cielab03022003.pdf. [Accessed: 15- Dec- 2017].

# Appendix 1 - Seven test images for iTMO

Figure 1 Chair

Figure 2 Car

Figure 3  Lake

Figure 4 Family

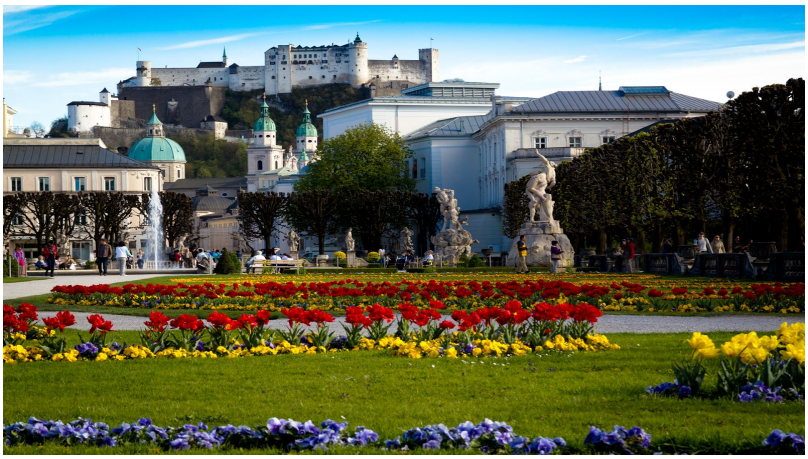


Figure 5 Garden

Figure 6 Home

Figure 7 Restaurant

# Appendix 2 - Matlab Code for iTMOs

```
clear all;
addpath(genpath('C:\Users\I856853\Desktop\EECE541\Project\Color_Correction_iTMO'));
% === Akyuz_iTMO ===
cd AkyuzOutput220;
for i=1:7
    ImgIn = imread(strcat('TestingFrames/',num2str(i),'.png'));
    ImgOut = Akyuz_iTMO( ImgIn , 4000 , 2.2);
    imwrite(ImgOut,strcat('AkyuzOutput',num2str(i),'.png'));
    %if need to save as HDR, go into Akyuz and comment out the last line
  %if save as .hdr, then use Q4 tool or picturenaut (free dl software)
end
cd ..;
%=== Banterle_iTMO ===
cd BanterleOutput;
for i=1:7
    ImgIn = imread(strcat('TestingFrames/',num2str(i),'.png'));
    ImgOut = Banterle_iTMO( ImgIn );
    imwrite(ImgOut,strcat('BanterleOutput',num2str(i),'.png'));

    end
cd ..;
%=== Bist_iTMO ===
cd BistOutput;
for i=1:7
    ImgIn = imread(strcat('TestingFrames/',num2str(i),'.png'));
    ImgOut = Bist_iTMO( ImgIn );
    imwrite(ImgOut,strcat('BistOutput',num2str(i),'.png'));
end
cd ..;
%=== Meylan_iTMO ===
cd MeylanOutput066;
for i=1:7
    ImgIn = imread(strcat('TestingFrames/',num2str(i),'.png'));
    ImgOut = Meylan_iTMO( ImgIn, 4000, 0.66 );
    imwrite(ImgOut,strcat('MeylanOutput',num2str(i),'.png'));
end
cd ..;
fprintf('It is Done');
```

# Appendix 3 - Matlab Code for Euclidean Distance Analysis

```
clear;
%Akyuz gamma=1.00
for i=1:7
    ImgIn = imread(strcat('AkyuzOutput100/AkyuzOutput',num2str(i),'.png'));
    ImgOut = imread(strcat('TestingFrames/',num2str(i),'.png'));
    dist(i,1)= sqrt(sum((ImgIn(:) - ImgOut(:)) .^ 2));
end
%Akyuz gamma=0.45
for i=1:7
    ImgIn = imread(strcat('AkyuzOutput045/AkyuzOutput',num2str(i),'.png'));
    ImgOut = imread(strcat('TestingFrames/',num2str(i),'.png'));
    dist(i,2)= sqrt(sum((ImgIn(:) - ImgOut(:)) .^ 2));
end
%Akyuz gamma=2.20
for i=1:7
    ImgIn = imread(strcat('AkyuzOutput220/AkyuzOutput',num2str(i),'.png'));
    ImgOut = imread(strcat('TestingFrames/',num2str(i),'.png'));
    dist(i,3)= sqrt(sum((ImgIn(:) - ImgOut(:)) .^ 2));
end
%Banterle
for i=1:7
    ImgIn = imread(strcat('BanterleOutput/BanterleOutput',num2str(i),'.png'));
    ImgOut = imread(strcat('TestingFrames/',num2str(i),'.png'));
    dist(i,4)= sqrt(sum((ImgIn(:) - ImgOut(:)) .^ 2));
end
%Bist
for i=1:7
    ImgIn = imread(strcat('BistOutput/BistOutput',num2str(i),'.png'));
    ImgOut = imread(strcat('TestingFrames/',num2str(i),'.png'));
    dist(i,5)= sqrt(sum((ImgIn(:) - ImgOut(:)) .^ 2));
end
%Meylan p=0.66
for i=1:7
    ImgIn = imread(strcat('MeylanOutput066/MeylanOutput',num2str(i),'.png'));
    ImgOut = imread(strcat('TestingFrames/',num2str(i),'.png'));
    dist(i,6)= sqrt(sum((ImgIn(:) - ImgOut(:)) .^ 2));
end
%Meylan p=0.33
for i=1:7
    ImgIn = imread(strcat('MeylanOutput033/MeylanOutput',num2str(i),'.png'));
    ImgOut = imread(strcat('TestingFrames/',num2str(i),'.png'));
    dist(i,7)= sqrt(sum((ImgIn(:) - ImgOut(:)) .^ 2));
end
```

# Appendix 4 - Matlab Code for Alpha Map Visualization

```
function varargout = SlideForLumiance(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @SlideForLumiance_OpeningFcn, ...
                   'gui_OutputFcn',  @SlideForLumiance_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end


function SlideForLumiance_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);
varargout{1} = handles.output;
global L
L=[0:20:100];
global Lindex
Lindex=2;%1-6
global B
B=[-100:20:100];
global Bindex%1-11
Bindex=2;
global A
A=[-100:20:100];
global Aindex%1-11
Aindex=2;
global ABEN
ABEN=[-100:19:90];


set(handles.edit1,'string',num2str(L(Lindex)));
set(handles.edit2,'string',num2str(B(Bindex)));
set(handles.edit3,'string',num2str(A(Aindex)));



% --- Executes on slider movement.
```

```
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)%%L-button
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Lindex
global L
global B
global A
global ABEN
global Aindex
global Bindex
global option %choose the ITMO
Lindex=Lindex-1;
if get(handles.radiobutton1,'value')
    option=1; %akyuz
elseif get(handles.radiobutton2,'value')
    option=2;%bentarle
elseif get(handles.radiobutton3,'value')
    option=3;%bist
elseif get(handles.radiobutton4,'value')
    option=4;%maylan
end
if (Lindex>=1&&Lindex<=6)
    set(handles.edit1,'string',num2str(L(Lindex)));
    %plot(handles.axes1,A,L(A),'r*');%testing code

    switch option
        case 1
            load('alphaMap_Akyuz.mat');
            AB=zeros(11,11);
            R=zeros(11,11);
            for i=1:11
                for j=1:11
                    AB(i,j)=alphaMap(L(Lindex)+1,A(i)+101,B(j)+101);
                    R(i,j)=determineRegion(L(Lindex),A(i),B(j));
                end
            end
            axes(handles.axes1);
            [X,Y]=meshgrid(A,B);
             mesh(X,Y,AB);
              hidden off;
hold on;
for i=1:11
                for j=1:11

                   switch R(i,j)
```

```
                    case 1
                plot3(A(i),B(j),0,'*m');
                    case 2
                        plot3(A(i),B(j),0,'pr');
                    case 3
                        plot3(A(i),B(j),0,'+y');
                    case 4
                        plot3(A(i),B(j),0,'<g');
                    case 5
                        plot3(A(i),B(j),0,'sc');
                    case 6
                        plot3(A(i),B(j),0,'>b');
                  end
               end
end
            hold off
            xlabel('A');
            ylabel('B');
            zlabel('alpha');

        case 2
           load('alphaMap_Banterle.mat');
            AB=zeros(11,11);
            R=zeros(11,11);
            for i=1:11
                for j=1:11
                    AB(i,j)=alphaMap(L(Lindex)+1,ABEN(i)+101,B(j)+101);
                    R(i,j)=determineRegion(L(Lindex),ABEN(i),B(j));
                end
            end
             axes(handles.axes1);
            [X,Y]=meshgrid(ABEN,B);
             mesh(X,Y,AB);
              hidden off;
hold on;
for i=1:11
               for j=1:11

                  switch R(i,j)
                      case 1
                          plot3(ABEN(i),B(j),0,'*m');
                      case 2
                          plot3(ABEN(i),B(j),0,'pr');
                      case 3
                          plot3(ABEN(i),B(j),0,'+y');
                      case 4
                          plot3(ABEN(i),B(j),0,'<g');
                      case 5
                          plot3(ABEN(i),B(j),0,'sc');
                      case 6
                          plot3(ABEN(i),B(j),0,'>b');
                  end
               end
```

```
end
            hold off
            xlabel('A');
            ylabel('B');
            zlabel('alpha');

        case 3
            load('alphaMap_Bist.mat');
            R=zeros(11,11);
            AB=zeros(11,11);
            for i=1:11
                for j=1:11
                    AB(i,j)=alphaMap(L(Lindex)+1,A(i)+101,B(j)+101);
                    R(i,j)=determineRegion(L(Lindex),A(i),B(j));
                end
            end
            axes(handles.axes1);
            [X,Y]=meshgrid(A,B);
             mesh(X,Y,AB);
             hidden off;
hold on;
for i=1:11
                for j=1:11

                   switch R(i,j)
                       case 1
                           plot3(A(i),B(j),0,'*m');
                       case 2
                           plot3(A(i),B(j),0,'pr');
                       case 3
                           plot3(A(i),B(j),0,'+y');
                       case 4
                           plot3(A(i),B(j),0,'<g');
                       case 5
                           plot3(A(i),B(j),0,'sc');
                       case 6
                           plot3(A(i),B(j),0,'>b');
                   end
                end
end
            hold off
            xlabel('A');
            ylabel('B');
            zlabel('alpha');


        case 4
             %meshz(handles.axes1,a,b,alpha);
%              xlabel('A');
%             ylabel('B');
%             zlabel('alpha');

    end
else
```

```
    set(handles.edit1,'string','illegal L');

end

function pushbutton3_Callback(hObject, eventdata, handles)
global Lindex
global L
global B
global A
global ABEN
global Aindex
global Bindex
global option
Lindex=Lindex+1;
if get(handles.radiobutton1,'value')
    option=1;
elseif get(handles.radiobutton2,'value')
    option=2;
elseif get(handles.radiobutton3,'value')
    option=3;
elseif get(handles.radiobutton4,'value')
    option=4;
end
if (Lindex>=1&&Lindex<=6)
    set(handles.edit1,'string',num2str(L(Lindex)));
    %plot(handles.axes1,A,L(A),'r*');%testing code
    switch option
        case 1
              load('alphaMap_Akyuz.mat');
            AB=zeros(11,11);
             R=zeros(11,11);
            for i=1:11
                for j=1:11
                    AB(i,j)=alphaMap(L(Lindex)+1,A(i)+101,B(j)+101);
                    R(i,j)=determineRegion(L(Lindex),A(i),B(j));
                end
            end

            [X,Y]=meshgrid(A,B);
            axes(handles.axes1);
             mesh(X,Y,AB);
             hidden off;
hold on;
for i=1:11
                for j=1:11

                   switch R(i,j)
                       case 1
                           plot3(A(i),B(j),0,'*m');
                       case 2
                           plot3(A(i),B(j),0,'pr');
                       case 3
                           plot3(A(i),B(j),0,'+y');
```

```
                    case 4
                        plot3(A(i),B(j),0,'<g');
                    case 5
                        plot3(A(i),B(j),0,'sc');
                    case 6
                        plot3(A(i),B(j),0,'>b');
                end
            end
end
            hold off
            xlabel('A');
            ylabel('B');
            zlabel('alpha');
        case 2
             load('alphaMap_Banterle.mat');
            AB=zeros(11,11);
             R=zeros(11,11);
            for i=1:11
                for j=1:11
                    AB(i,j)=alphaMap(L(Lindex)+1,ABEN(i)+101,B(j)+101);
                    R(i,j)=determineRegion(L(Lindex),ABEN(i),B(j));
                end
            end
              axes(handles.axes1);
            [X,Y]=meshgrid(ABEN,B);
             mesh(X,Y,AB);
             hidden off;
hold on;
for i=1:11
            for j=1:11

                switch R(i,j)
                    case 1
                        plot3(ABEN(i),B(j),0,'*m');
                    case 2
                        plot3(ABEN(i),B(j),0,'pr');
                    case 3
                        plot3(ABEN(i),B(j),0,'+y');
                    case 4
                        plot3(ABEN(i),B(j),0,'<g');
                    case 5
                        plot3(ABEN(i),B(j),0,'sc');
                    case 6
                        plot3(ABEN(i),B(j),0,'>b');
                end
            end
end
            hold off
            xlabel('A');
            ylabel('B');
            zlabel('alpha');
        case 3
            load('alphaMap_Bist.mat');
              R=zeros(11,11);
```

```
            AB=zeros(11,11);
            for i=1:11
                for j=1:11
                    AB(i,j)=alphaMap(L(Lindex)+1,A(i)+101,B(j)+101);
                     R(i,j)=determineRegion(L(Lindex),A(i),B(j));
                end
            end

            [X,Y]=meshgrid(A,B);
             axes(handles.axes1);
             mesh(X,Y,AB);
              hidden off;
hold on;
for i=1:11
                for j=1:11

                   switch R(i,j)
                       case 1
                           plot3(A(i),B(j),0,'*m');
                       case 2
                           plot3(A(i),B(j),0,'pr');
                       case 3
                           plot3(A(i),B(j),0,'+y');
                       case 4
                           plot3(A(i),B(j),0,'<g');
                       case 5
                           plot3(A(i),B(j),0,'sc');
                       case 6
                           plot3(A(i),B(j),0,'>b');
                   end
                end
end
            hold off
            xlabel('A');
            ylabel('B');
            zlabel('alpha');
        case 4
%              %meshz(handles.axes1,a,b,alpha);
%              xlabel('A');
%             ylabel('B');
%             zlabel('alpha');

    end
else
    set(handles.edit1,'string','illegal L, please set L to 0-100');
end




function edit1_Callback(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

set(handles.radiobutton1,'value',1);
set(handles.radiobutton2,'value',0);
set(handles.radiobutton3,'value',0);
set(handles.radiobutton4,'value',0);

function radiobutton2_Callback(hObject, eventdata, handles)

set(handles.radiobutton1,'value',0);
set(handles.radiobutton2,'value',1);
set(handles.radiobutton3,'value',0);
set(handles.radiobutton4,'value',0);



% --- Executes on button press in radiobutton3.
function radiobutton3_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.radiobutton1,'value',0);
set(handles.radiobutton2,'value',0);
set(handles.radiobutton3,'value',1);
set(handles.radiobutton4,'value',0);


% Hint: get(hObject,'Value') returns toggle state of radiobutton3


% --- Executes on button press in radiobutton4.
function radiobutton4_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.radiobutton1,'value',0);
set(handles.radiobutton2,'value',0);
set(handles.radiobutton3,'value',0);
set(handles.radiobutton4,'value',1);


% Hint: get(hObject,'Value') returns toggle state of radiobutton4


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO) A-button plot gif in axes2
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
global Lindex
global L
global B
global A
global ABEN

global Aindex
global Bindex
global option
Aindex=Aindex-1;

if get(handles.radiobutton1,'value')
    option=1;
elseif get(handles.radiobutton2,'value')
    option=2;
elseif get(handles.radiobutton3,'value')
    option=3;
elseif get(handles.radiobutton4,'value')
    option=4;
end
if (Aindex>=1&&Aindex<=11)
    set(handles.edit2,'string',num2str(A(Aindex)));
    %plot(handles.axes1,A,L(A),'r*');%testing code
    switch option
        case 1
            set(handles.edit2,'string',num2str(A(Aindex)));
           load('alphaMap_Akyuz.mat');
           Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,A(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');

        case 2
            set(handles.edit2,'string',num2str(ABEN(Aindex)));
            load('alphaMap_Banterle.mat');
             Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,ABEN(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 3
           set(handles.edit2,'string',num2str(A(Aindex)));
            load('alphaMap_Bist.mat');
           Lmap=zeros(1,6);
```

```
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,A(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 4
            set(handles.edit2,'string',num2str(A(Aindex)));
%meshz(handles.axes1,a,b,alpha);
%              xlabel('A');
%             ylabel('B');
%             zlabel('alpha');

    end
else
    set(handles.edit2,'string','illegal A');
end


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO) A+button
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Lindex
global L
global B
global A
global ABEN
global Aindex
global Bindex
global option
Aindex=Aindex+1;
if get(handles.radiobutton1,'value')
    option=1;
elseif get(handles.radiobutton2,'value')
    option=2;
elseif get(handles.radiobutton3,'value')
    option=3;
elseif get(handles.radiobutton4,'value')
    option=4;
end
if (Aindex>=1&&Aindex<=11)
    set(handles.edit2,'string',num2str(A(Aindex)));
    %plot(handles.axes1,A,L(A),'r*');%testing code
    switch option
        case 1
           set(handles.edit2,'string',num2str(A(Aindex)));
            load('alphaMap_Akyuz.mat');
           Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,A(Aindex)+101,B(Bindex)+101);
```

```
            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 2
            set(handles.edit2,'string',num2str(ABEN(Aindex)));
            load('alphaMap_Banterle.mat');
             Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,ABEN(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 3
            set(handles.edit2,'string',num2str(A(Aindex)));
            load('alphaMap_Bist.mat');
           Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,A(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 4
              set(handles.edit2,'string',num2str(A(Aindex)));
%meshz(handles.axes1,a,b,alpha);
%              xlabel('A');
%             ylabel('B');
%             zlabel('alpha');

    end
else
    set(handles.edit2,'string','illegal A');
end



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double


% --- Executes during object creation, after setting all properties.
```

```
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)%%B- button
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Lindex
global L
global B
global A
global ABEN
global Aindex
global Bindex
global option
Bindex=Bindex-1;
if get(handles.radiobutton1,'value')
    option=1;
elseif get(handles.radiobutton2,'value')
    option=2;
elseif get(handles.radiobutton3,'value')
    option=3;
elseif get(handles.radiobutton4,'value')
    option=4;
end
if (Bindex>=1&&Bindex<=11)
    set(handles.edit3,'string',num2str(B(Bindex)));
    %plot(handles.axes1,A,L(A),'r*');%testing code
    switch option
        case 1
         load('alphaMap_Akyuz.mat');
           Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,A(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 2
            load('alphaMap_Banterle.mat');
             Lmap=zeros(1,6);
```

```
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,ABEN(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 3
            load('alphaMap_Bist.mat');
           Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,A(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 4
%               %meshz(handles.axes1,a,b,alpha);
%               xlabel('A');
%              ylabel('B');
%              zlabel('alpha');

    end
else
    set(handles.edit3,'string','illegal B');
end


% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles)
global Lindex
global L
global B
global A
global ABEN
global Aindex
global Bindex
global option
Bindex=Bindex+1;
if get(handles.radiobutton1,'value')
    option=1;
elseif get(handles.radiobutton2,'value')
    option=2;
elseif get(handles.radiobutton3,'value')
    option=3;
elseif get(handles.radiobutton4,'value')
    option=4;
end
if (Bindex>=1&&Bindex<=11)
    set(handles.edit3,'string',num2str(B(Bindex)));
    %plot(handles.axes1,A,L(A),'r*');%testing code
```

```
    switch option
        case 1
            load('alphaMap_Akyuz.mat');
           Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,A(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
         plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 2
            load('alphaMap_Banterle.mat');
             Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,ABEN(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 3
            load('alphaMap_Bist.mat');
           Lmap=zeros(1,6);
            for i=1:6
                    Lmap(i)=alphaMap(L(i)+1,A(Aindex)+101,B(Bindex)+101);

            end
            axes(handles.axes2);
            plot(L,Lmap,'r-o');
            xlabel('L');
            ylabel('alpha');
        case 4
%              %meshz(handles.axes1,a,b,alpha);

    end
else
    set(handles.edit3,'string','illegal B');
end



function edit3_Callback(hObject, eventdata, handles)

function edit3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

# Appendix 5 - Matlab Code for Color Region Determination

```
function [regionYcbcr]=determineRegion(l,a,b)
lab(1,1,1)=l;
lab(1,1,2)=a;
lab(1,1,3)=b;
rgb=lab2rgb(lab,'OutputType','uint8');
ycbcr=rgb2ycbcr(rgb);
cb=double(ycbcr(1,1,2));
cr=double(ycbcr(1,1,3));
theta=atand(cr/cb);
if(theta<0)
    theta=360+theta;
end
if(theta>22&&theta<=80)
    regionYcbcr=1;%magenta
elseif(theta>80&&theta<=138)
     regionYcbcr=2;%red
    elseif(theta>138&&theta<=202)
     regionYcbcr=3;%yellow
    elseif(theta>202&&theta<=259)
     regionYcbcr=4;%green
elseif(theta>259&&theta<=319)
      regionYcbcr=5;%cyan
else
      regionYcbcr=6;%blue
   end

end
```

# Appendix 6 - Matlab Code for DeltaE

```
function deltaE = deltaELab(imgSDR,imgHDR)
%deltaELab - calculate the deltaE value in Lab for two images
%
% Syntax:  deltaE = deltaELab(imgSDR,imgHDR)
%
%---------------------------- BEGIN CODE ---------------------------------
imgSDR_double = double(imgSDR)./255;
imgSDR_GammaRemoved = imgSDR_double.^2.2;
imgSDR_XYZ = RGB2XYZ(imgSDR_GammaRemoved);
imgSDR_Lab = xyz2lab(imgSDR_XYZ);

imgHDR_double= double(imgHDR)./4000;
imgHDR_XYZ = RGB2XYZ(imgHDR_double);
imgHDR_Lab = xyz2lab(imgHDR_XYZ);
%imgHDR_GammaRemoved = imgHDR_double.^2.2; %no need for HDR image

deltaE = sqrt(sum((imgSDR_Lab(:) - imgHDR_Lab(:)) .^ 2));
```

```
end
```

# Appendix 7 - Matlab Code for Main function of global alpha color correction method

```
%alpha_deltaE_adjustment.m
%This code pick an iTMO and test alpha values that would result in the
%lowest deltaE value. The results are use to create the hdr image in both
%.png & .hdr formats.
clc;
clear all;
addpath(genpath('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes'));
%choose the iTMO - 1=Akyuz100; 2=Banterle; 3=Bist; 4=Meylan066
iTMONum=3;
dir='C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB Codes';
switch iTMONum
    case 1
        iTMOName='Akyuz100';
        dirName='C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz';
        fileName='AkyuzOutput100Adjusted';
    case 2
        iTMOName='Banterle';
        dirName='C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Banterle';
        fileName='BanterleOutputAdjusted';
    case 3
        iTMOName='Bist';
        dirName='C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes';
        fileName='BistOutputAdjusted';
    case 4
        iTMOName='Meylan066';

dirName='C:\Users\I856853\Desktop\EECE541\Project\Color_Correction_iTMO\MeylanOutput0
66';
        fileName='MeylanOutput066Adjusted';
end
deltaE=zeros(1,7);
for i=3:3
     %load SDR image
    ImgSDR =
imread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\',num2str(i),'.png'));
    %loading HDR image
    switch iTMONum
    case 1
        ImgHDR=Akyuz_iTMO( ImgSDR , 4000 , 1
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\AkyuzOutput',num2str(i),'.hdr'));
```

```
    case 2
        imgHDR=Banterle_iTMO( imageSDR
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\BanterleOutput',num2str(i),'.hdr'));
    case 3
        ImgHDR=Bist_iTMO( ImgSDR
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\BistOutput',num2str(i),'.hdr'));
    case 4
        imgHDR=Meylan_iTMO( imageSDR );
    end

    %create hdr img with alpha value that has the lowest deltaE
    [alphaOut,deltaE_min,alpha,deltaE,ImgHDR_adjusted]=
findAlphaOutputHdrRev3(ImgSDR,ImgHDR,0,10,0.1);
    %create .hdr & .png files
    cd(dir);
    %hdrwrite(ImgHDR_adjusted,strcat(fileName,'Global',num2str(i),'.hdr'));
   % ImgHDR_Sim2 =
Sim2ShaderTechnicolor(ImgHDR_adjusted(:,:,1),ImgHDR_adjusted(:,:,2),ImgHDR_adjusted(:
,:,3));
    %imwrite(ImgHDR_Sim2,strcat(fileName,'Global',num2str(i),'.png'));
    cd ..;
imgSDR_double= double(ImgSDR)./255;
imgSDR_GammaRemoved = imgSDR_double.^2.2;
imgSDR_XYZ = rgb2xyz(imgSDR_GammaRemoved );
imgSDR_LAB= xyz2lab(imgSDR_XYZ);

imgHDR_double= (double(ImgHDR_adjusted)-0.005)./(4000-0.0005);
imgHDR_XYZ = rgb2xyz(imgHDR_double);
imgHDR_LAB= xyz2lab(imgHDR_XYZ);
%2. calculate deltaE
deltaE(1,i)=calculateDeltaE(imgSDR_LAB,imgHDR_LAB);
deltaE(1,i)=deltaE_min;
end
%xlswrite('deltaE1.xlsx', deltaE, 'Sheet2', 'A1:G2');
```

# Appendix 8 - Matlab Code for Main function of local alpha color correction method

```
clc;
clear all;
tic;
iTMONum=1;
switch iTMONum
    case 1
       load alphaMap_AkyuzStep5.mat;
    case 2
        load alphaMap_Banterle.mat;
    case 3
```

```
        load alphaMap_Bist.mat;
    case 4
       load alphaMap_Akyuz.mat;
end
%alphaMap(alphaMap==10)=1;
deltaE=zeros(2,7);%two rows represent deltaE of original HDR and color corrected HDR
respectively

for i=1:7
    %original SDR image
imageSDR=imread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\image
\',num2str(i),'.png'));
%HDR image after iTMO
switch iTMONum
    case 1
        imageHDR=Akyuz_iTMO( imageSDR , 4000 , 1
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\AkyuzOutput',num2str(i),'.hdr'));
    case 2
        imageHDR=Banterle_iTMO( imageSDR
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\BanterleOutput',num2str(i),'.hdr'));
    case 3
        imageHDR=Bist_iTMO( imageSDR
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\BistOutput',num2str(i),'.hdr'));
    case 4
        imageHDR=Meylan_iTMO( imageSDR
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\BistOutput',num2str(i),'.hdr'));
end
%output after color correction
 imgOut= findsdrpixelsnew(imageSDR,imageHDR,alphaMap);

%hdrwrite(imgOut,strcat('AllPixCCForiTMO',num2str(iTMONum),'Pic',num2str(i),'Step5.hd
r'));
 %ImgHDR_Sim2 = Sim2ShaderTechnicolor(imgOut(:,:,1),imgOut(:,:,2),imgOut(:,:,3));
%
imwrite(ImgHDR_Sim2,strcat('AllPixCCForiTMO',num2str(iTMONum),'Pic',num2str(i),'Step5
.png'));

%calculate deltaE for original HDR and color corrected HDR
%1. transform rgb to lab for sdr, orig hdr and cc hdr
imgSDR_double= double(imageSDR)./255;
imgSDR_GammaRemoved = imgSDR_double.^2.2;
imgSDR_XYZ = rgb2xyz(imgSDR_GammaRemoved );
imgSDR_LAB= xyz2lab(imgSDR_XYZ);

imgHDR_double= (double(imageHDR)-0.005)./(4000-0.0005);
imgHDR_XYZ = rgb2xyz(imgHDR_double);
oriHDR_LAB= xyz2lab(imgHDR_XYZ);

imgHDR_double= (double(imgOut)-0.005)./(4000-0.0005);
imgHDR_XYZ = rgb2xyz(imgHDR_double);
```

```
ccHDR_LAB= xyz2lab(imgHDR_XYZ);
%2. calculate deltaE
deltaE(1,i)=calculateDeltaE(imgSDR_LAB,oriHDR_LAB);
deltaE(2,i)=calculateDeltaE(imgSDR_LAB,ccHDR_LAB);
end
xlswrite('deltaE1.xlsx', deltaE, 'Sheet1', 'A11:G12');
Toc;
```

# Appendix 9 - Matlab Code for alphaMap generation

```
%alphaMapRev4.m
%this code test combinations of LAB values to find their corresponding
%alpha value that can result in the lowest deltaE
clc;
clear all;
tic;
%select iTMO
iTMO=3; %1: Akyuz; 2: Banterle; 3:Bist; 4:Meylan
%define LAB values test ranges
switch(iTMO)
    case 1 %Akyuz
        LStart=0; LStep=20; LEnd=100;
        AStart=-100; AStep=20; AEnd=100;
        BStart=-100; BStep=20; BEnd=100;
        alphaTestRange=10;
    case 2 %Banterle
        LStart=0; LStep=20; LEnd=100;
        AStart=-100; AStep=19; AEnd=90;
        BStart=-100; BStep=20; BEnd=100;
        alphaTestRange=100;
    case 3 %Bist
        LStart=0; LStep=5; LEnd=100;
        AStart=-100; AStep=5; AEnd=100;
        BStart=-100; BStep=5; BEnd=100;
        alphaTestRange=10;
    case 4 %Meylan
        LStart=0; LStep=20; LEnd=100;
        AStart=-100; AStep=19; AEnd=90;
        BStart=-100; BStep=20; BEnd=100;
        alphaTestRange=100;
end
alphaMap=zeros([100 200 200]);
%try 5*10*10 = 500 LAB values
for L=LStart:LStep:LEnd %set steps to 20 to reduce steps when debugging
    for A=AStart:AStep:AEnd
        for B=BStart:BStep:BEnd
            %create image
            for i=1:10
                for j=1:10
                    img(i,j,1)=L;
                    img(i,j,2)=A;
```

```
                    img(i,j,3)=B;
                end
            end
%             img(1,1,1)=L;
%             img(1,1,2)=A;
%             img(1,1,3)=B;
            %convert img from Lab to RGB
            imgNormalized=lab2rgb(img);
            imgNormalized(imgNormalized<0)=0;
            % add gamma
            imgNormalizedGamma=imgNormalized.^(1/2.2);
            %scale to range 0 to 255
            imgSDR=imgNormalizedGamma*255;
            %perform iTMO
            switch(iTMO)
                case 1 %Akyuz
                    imgHDR  = Akyuz_iTMO_hdr( imgSDR , 4000 , 1);
                case 2 %Banterle
                    imgHDR = Banterle_iTMO_hdr(imgSDR);
                case 3 %Bist
                    imgHDR  = Bist_iTMO( imgSDR);
                case 4 %Meylan
                    imgHDR = Meylan_iTMO_hdr( imgSDR, 4000, 0.66 );
            end
            %find alpha with the lowest deltaE value
            [alphaOut,deltaE_min,alpha,deltaE]
=findAlpha(imgSDR,imgHDR,0,alphaTestRange,0.1);
            % store value in 3D matrix
            L_=L+1;
            A_=A+101;
            B_=B+101;
            alphaMap(L_,A_,B_)=alphaOut;
            if (iTMO==1 && L==0 && A==0 && B==0)
                alphaMap(L_,A_,B_)=1;
            end
            %debug code
            if (alphaOut==0)
                disp(alphaOut);
            end
            txt=['L: ',num2str(L),'; A: ',num2str(A),'; B: ',num2str(B),'; alphaOut:
',num2str(alphaOut)];
            disp(txt);
        end
    end
end
toc%output time elapsed
txt=['Max alpha: ',num2str(max(max(max(alphaMap))))];
```

# Appendix 10 - Matlab Code for finding the alpha corresponding to the minimum deltaE

```
%function [imgOut,alphaOut]= findAlphaOutputHdr(
imageSDR,imageHDR,startVal,endVal,stepVal)
function [alphaOut,deltaE_min,alpha,deltaE,imgOut] = findAlphaOutputHdr(
imageSDR,imageHDR,startVal,endVal,stepVal)
%this function takes SDR images and HDR images and find
%returns rgb values alpha values with the least deltaE value
%
%        Input:
%           -imageSDR: input SDR image from imread
%           -imageHDR: input HDR image from hdrread
%           -startVal: alpha value to start testing
%           -endVal: alpha value to end testing
%           -stepSize: step size of alpha value
%
%        Output:
%           -imgOut: hdr image in rgb colour space to be used with imwrite
%           or hdrwrite
%
%create test values
alpha=[startVal:stepVal:endVal];
deltaE=[];
%normalize and convert HDR image to Lab Colour space
imgHDR_double= (double(imageHDR)-0.005)./(4000-0.0005);
imgHDR_XYZ = rgb2xyz(imgHDR_double);
L_HDR_XYZ=max(max(imgHDR_XYZ(:,:,2)));
imgHDR_LAB= xyz2lab(imgHDR_XYZ);
%normalize, remove gamma, and convert SDR image to Lab Colour space
imgSDR_double= double(imageSDR)./255;
imgSDR_GammaRemoved = imgSDR_double.^2.2;
imgSDR_XYZ = rgb2xyz(imgSDR_GammaRemoved );
L_SDR_XYZ=max(max(imgSDR_XYZ(:,:,2)));
imgSDR_LAB= xyz2lab(imgSDR_XYZ);
%extract Luminance L from SDR image
L_SDR=imgSDR_LAB(:,:,1);
L_SDR_max=max(max(L_SDR));
%extract chrominance a from SDR image
A_SDR=imgSDR_LAB(:,:,2);
%extract chrominance b from SDR image
B_SDR=imgSDR_LAB(:,:,3);
%extract Luminance L from HDR image
L_HDR=imgHDR_LAB(:,:,1);
L_HDR_max=max(max(L_HDR));
%extract chrominance a from HDR image
A_HDR=imgHDR_LAB(:,:,2);
%extract chrominance b from HDR image
B_HDR=imgHDR_LAB(:,:,3);
%scale L_HDR and L_SDR into [0,1]
L_HDR_norm=L_HDR./100;
```

```
L_SDR_norm=L_SDR./100;
L_SDR_norm(find(L_SDR_norm==0))=0.001;
%calculate the ratio with luminance of HDRI and SDRI
LRatio=L_HDR_norm./L_SDR_norm;
ratio_max=max(max(LRatio));
%testing with all alpha values
for i=1:((endVal-startVal)/stepVal+1)
    imgHDR_LAB(:,:,2)= A_SDR .*LRatio*alpha(i);
    imgHDR_LAB(:,:,3)= B_SDR .*LRatio*alpha(i);
    %imgHDR_LAB(:,:,2)= A_HDR .*alpha(i);
    %imgHDR_LAB(:,:,3)= B_HDR .*alpha(i);
    deltaE(i)=sqrt(sum((imgHDR_LAB(:) - double(imgSDR_LAB(:))) .^ 2));
end
%find alpha value that results in the min deltaE value
deltaE_min=min(deltaE);
alphaOut=alpha(find(min(deltaE)==deltaE));
%scale the the a & b of SDR image with the alpha value
imgHDR_LAB(:,:,2)= A_SDR .*LRatio *alphaOut(1);
imgHDR_LAB(:,:,3)= B_SDR .*LRatio *alphaOut(1);
%convert img from Lab to RGB
imgOutNormalized=lab2rgb(imgHDR_LAB);
imgOutNormalized(imgOutNormalized<0)=0;
%scale to range 0.005 to 4000
imgOut=imgOutNormalized*(4000-0.005)+0.005;
%imgOut(:,:,1)=imageHDR(:,:,1);
end
```

# Appendix 11 - Matlab Code for searching for the alpha in the alphaMap for images

```
function imgOut= findsdrpixelsnew(imageSDR,imageHDR,alphaMap)
%normalize, remove gamma, and convert SDR image to Lab Colour space
imgSDR_double= double(imageSDR)./255;
imgSDR_GammaRemoved = imgSDR_double.^2.2;
imgSDR_XYZ = rgb2xyz(imgSDR_GammaRemoved );
imgSDR_LAB= xyz2lab(imgSDR_XYZ);
%normalize HDR image and change it to lab
imgHDR_double= (double(imageHDR)-0.005)./(4000-0.0005);
imgHDR_XYZ = rgb2xyz(imgHDR_double);
%L_HDR_XYZ=max(max(imgHDR_XYZ(:,:,2)));
imgHDR_LAB= xyz2lab(imgHDR_XYZ);
bound=min(min(min(imgHDR_LAB)));

%get LAB value of SDRimage
     L=imgSDR_LAB(:,:,1);
     A=imgSDR_LAB(:,:,2);
     B=imgSDR_LAB(:,:,3);
     L_min=min(L(:));
     A_min=min(A(:));
     B_min=min(B(:));
```

```
for i = 1:1080
    for j = 1:1920
        %set l,a,b as the index of alphaMap to find alpha for l,a,b (when alpha==0,
choose the alpha of the closest point)
    l=round(double(L(i,j))/5)*5+1;
    a=round(double(A(i,j))/5)*5+101;
    b=round(double(B(i,j))/5)*5+101;
    if(l<=0)
        l=1;
    end
    if(a<=0)
        a=1;
    end
    if(b<=0)
        b=1;
    end

    alpha=alphaMap(l,a,b);
   imgHDR_LAB(i,j,2)=alpha*imgHDR_LAB(i,j,2);
   imgHDR_LAB(i,j,3)=alpha*imgHDR_LAB(i,j,3);
    end
end
%convert img from Lab to RGB
imgOutNormalized=lab2rgb(imgHDR_LAB);
imgOutNormalized(imgOutNormalized<0)=0;
%scale to range 0.005 to 4000
imgOut=imgOutNormalized*(4000-0.005)+0.005;

%newalpha= newalpha(:,:,:);
%dlmwrite('D:\541\Color_Correction_iTMO\MATLAB Codes\Needed
Functions\data.txt',newalpha);
end
```

# Appendix 12 - Matlab Code for drawing histogram based on the alphamap

```
iTMONum=1;clc;
clear all;

switch iTMONum
    case 1
       load alphaMap_AkyuzStep5.mat;
    case 2
        load alphaMap_Banterle.mat;
    case 3
        load alphaMap_Bist20.mat;
    case 4
       load alphaMap_Akyuz.mat;
end
count=1;
for l=1:5:100
```

```
for a=1:5:200
for b=1:5:200
alphaCollection(count)=alphaMap(l,a,b);
count=count+1;
end
end
end
alphaCollection=sort(alphaCollection);
alphafreq=hist(alphaCollection,100);
hist(alphaCollection,100);
%axis([0 10 0 35000]);
xlabel('alpha') % x-axis label
ylabel('frequency')
```

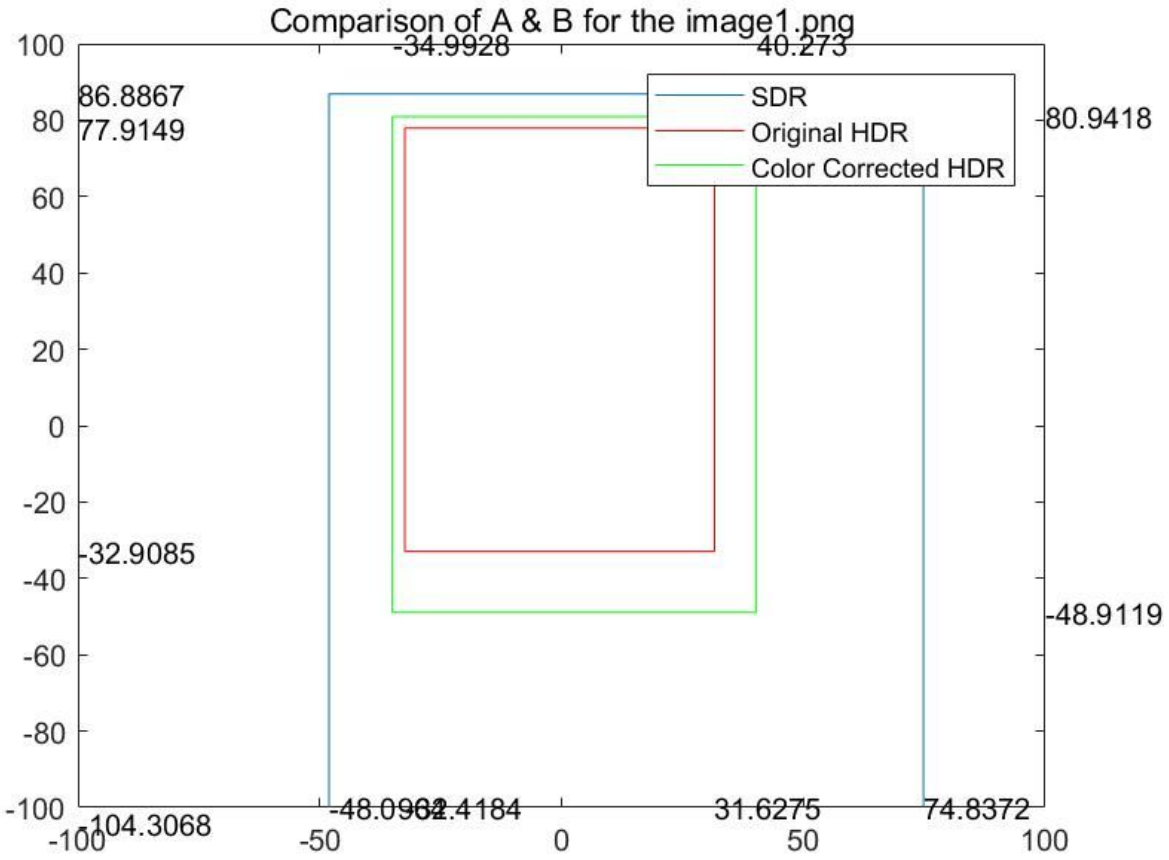
# Appendix 13 - Matlab Code for comparing the bound of A and B

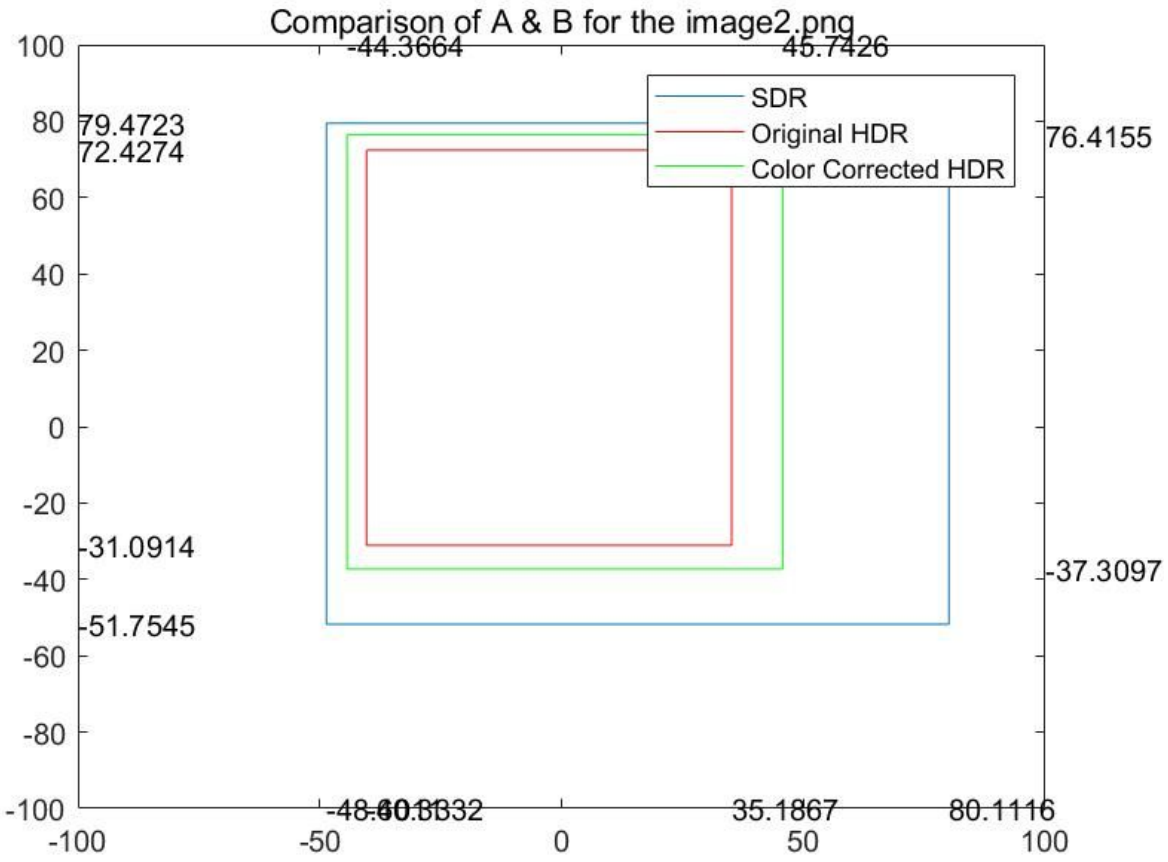
```
 clc;
clear all;
iTMONum=1;
switch iTMONum
    case 1
       load alphaMap_AkyuzStep5.mat;
    case 2
        load alphaMap_Banterle.mat;
    case 3
        load alphaMap_Bist.mat;
    case 4
       load alphaMap_Akyuz.mat;
end
for i=1:1
    %original SDR image
imageSDR=imread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\image
\',num2str(i),'.png'));
%HDR image after iTMO
switch iTMONum
    case 1
        imageHDR=Akyuz_iTMO( imageSDR , 4000 , 1
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\AkyuzOutput',num2str(i),'.hdr'));
    case 2
        imageHDR=Banterle_iTMO( imageSDR
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\BanterleOutput',num2str(i),'.hdr'));
    case 3
        imageHDR=Bist_iTMO( imageSDR
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\BistOutput',num2str(i),'.hdr'));
    case 4
```

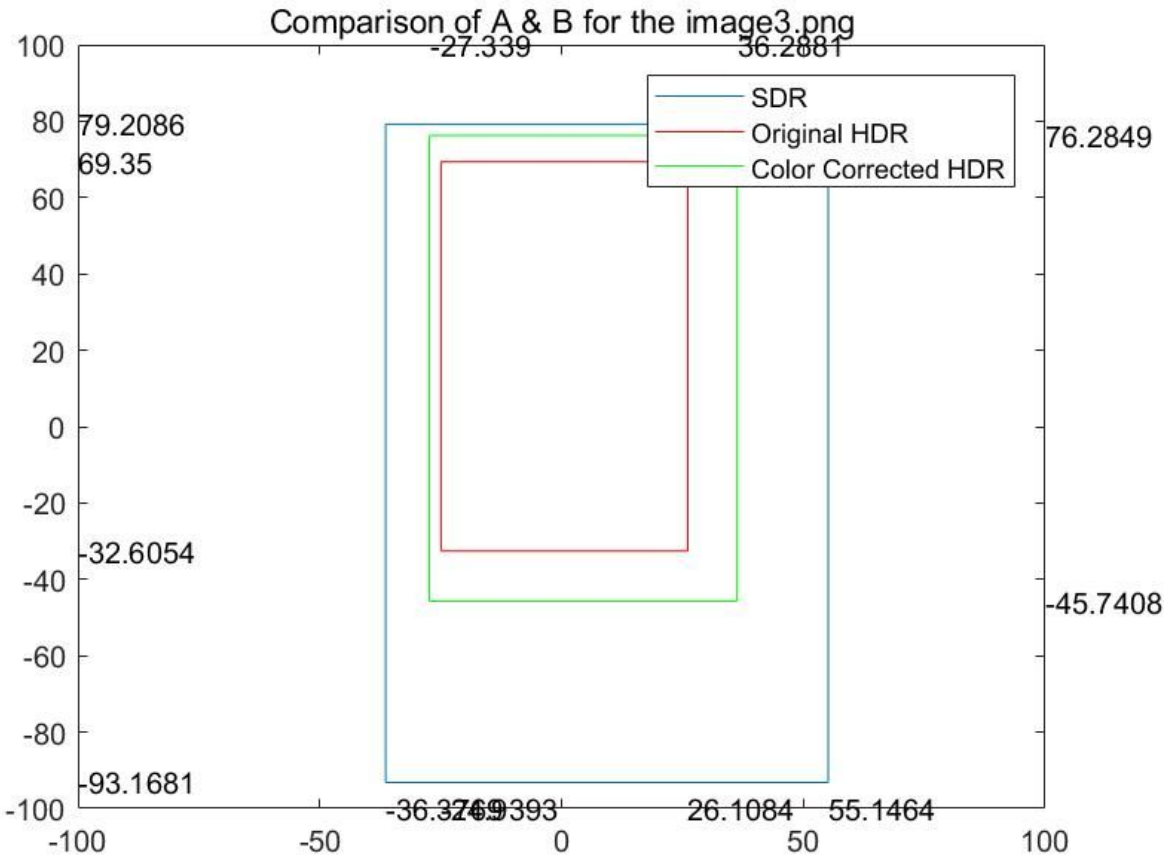
```
        imageHDR=Meylan_iTMO( imageSDR
);%hdrread(strcat('C:\Users\wangzhen\Desktop\eece541\Color_Correction_iTMO\MATLAB
Codes\Akyuz\AkyuzOutput100\BistOutput',num2str(i),'.hdr'));
end

imgOut= findsdrpixelsnew(imageSDR,imageHDR,alphaMap);

imgSDR_double= double(imageSDR)./255;
imgSDR_GammaRemoved = imgSDR_double.^2.2;
imgSDR_XYZ = rgb2xyz(imgSDR_GammaRemoved );
imgSDR_LAB= xyz2lab(imgSDR_XYZ);
a_SDR=imgSDR_LAB(:,:,2);
b_SDR=imgSDR_LAB(:,:,3);
[asdr_max,asdr_min]=findMaxMin(a_SDR);
[bsdr_max,bsdr_min]=findMaxMin(b_SDR);
pos=[asdr_max,asdr_min,asdr_min,asdr_max,asdr_max;bsdr_max,bsdr_max,bsdr_min,bsdr_min
,bsdr_max];
%subplot(3,3,i);
figure(i)

plot(pos(1,:),pos(2,:));
text(-100,bsdr_max,num2str(bsdr_max));
text(-100,bsdr_min,num2str(bsdr_min));
text(asdr_max,-100,num2str(asdr_max));
text(asdr_min,-100,num2str(asdr_min));
axis([-100 100 -100 100])
hold on

imgHDR_double= (double(imageHDR)-0.005)./(4000-0.0005);
imgHDR_XYZ = rgb2xyz(imgHDR_double);
oriHDR_LAB= xyz2lab(imgHDR_XYZ);
a_oriHDR=oriHDR_LAB(:,:,2);
b_oriHDR=oriHDR_LAB(:,:,3);
[aohdr_max,aohdr_min]=findMaxMin(a_oriHDR);
[bohdr_max,bohdr_min]=findMaxMin(b_oriHDR);
pos1=[aohdr_max,aohdr_min,aohdr_min,aohdr_max,aohdr_max;bohdr_max,bohdr_max,bohdr_min
,bohdr_min,bohdr_max];
plot(pos1(1,:),pos1(2,:),'r');
text(-100,bohdr_max,num2str(bohdr_max));
text(-100,bohdr_min,num2str(bohdr_min));
text(aohdr_max,-100,num2str(aohdr_max));
text(aohdr_min,-100,num2str(aohdr_min));
hold on

imgHDR_double= (double(imgOut)-0.005)./(4000-0.0005);
imgHDR_XYZ = rgb2xyz(imgHDR_double);
ccHDR_LAB= xyz2lab(imgHDR_XYZ);
a_ccHDR=ccHDR_LAB(:,:,2);
b_ccHDR=ccHDR_LAB(:,:,3);
[achdr_max,achdr_min]=findMaxMin(a_ccHDR);
[bchdr_max,bchdr_min]=findMaxMin(b_ccHDR);
pos2=[achdr_max,achdr_min,achdr_min,achdr_max,achdr_max;bchdr_max,bchdr_max,bchdr_min
,bchdr_min,bchdr_max];
plot(pos2(1,:),pos2(2,:),'g');
```

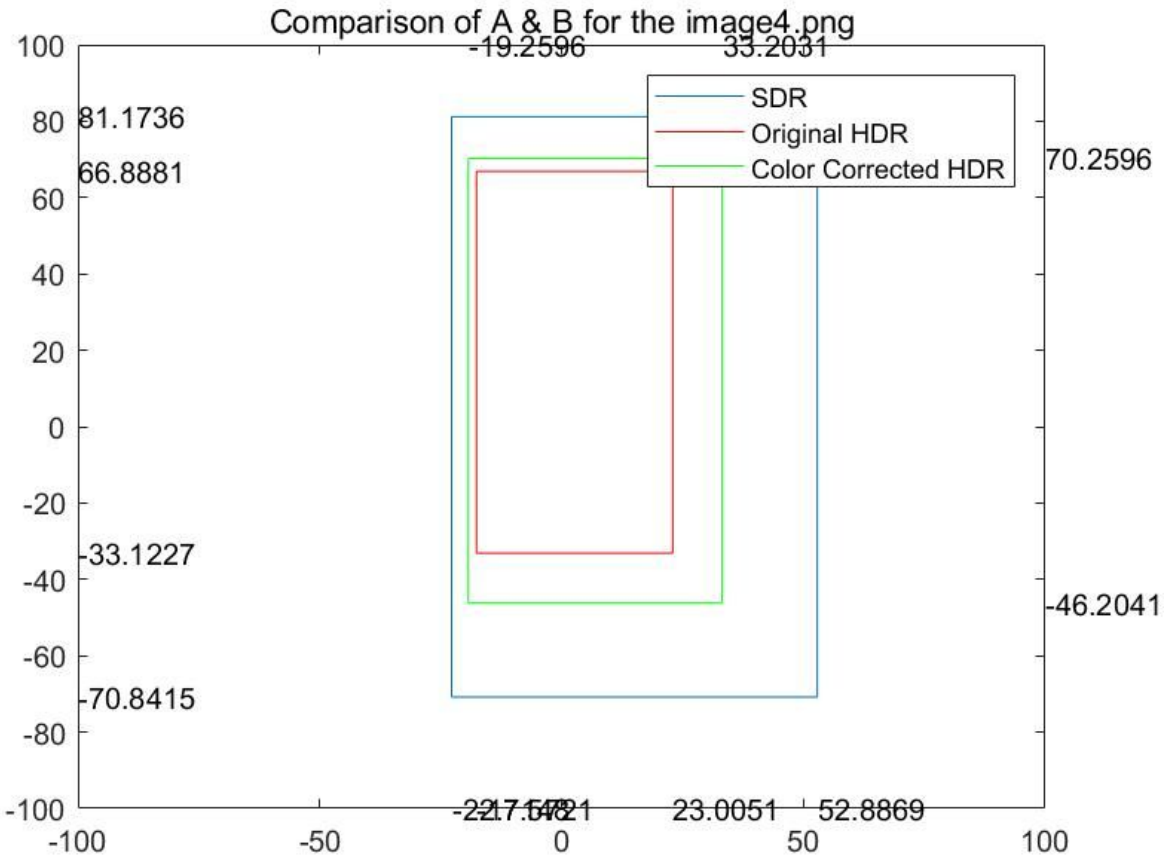
```
text(100,bchdr_max,num2str(bchdr_max));
text(100,bchdr_min,num2str(bchdr_min));
text(achdr_max,100,num2str(achdr_max));
text(achdr_min,100,num2str(achdr_min));
legend('SDR','Original HDR','Color Corrected HDR');
title(strcat('Comparison of A & B for the image ',num2str(i),'.png'));

End
```

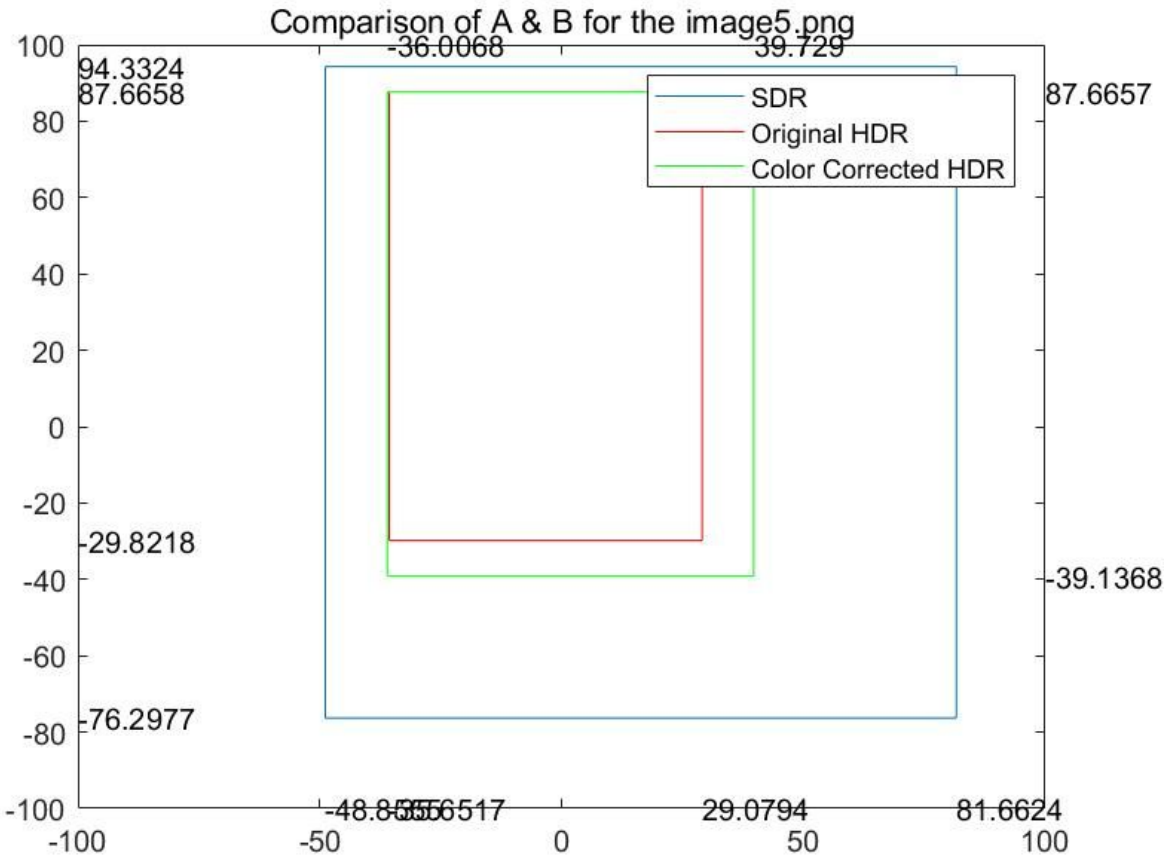
# Appendix 14 - Additional figures of comparison between A & B

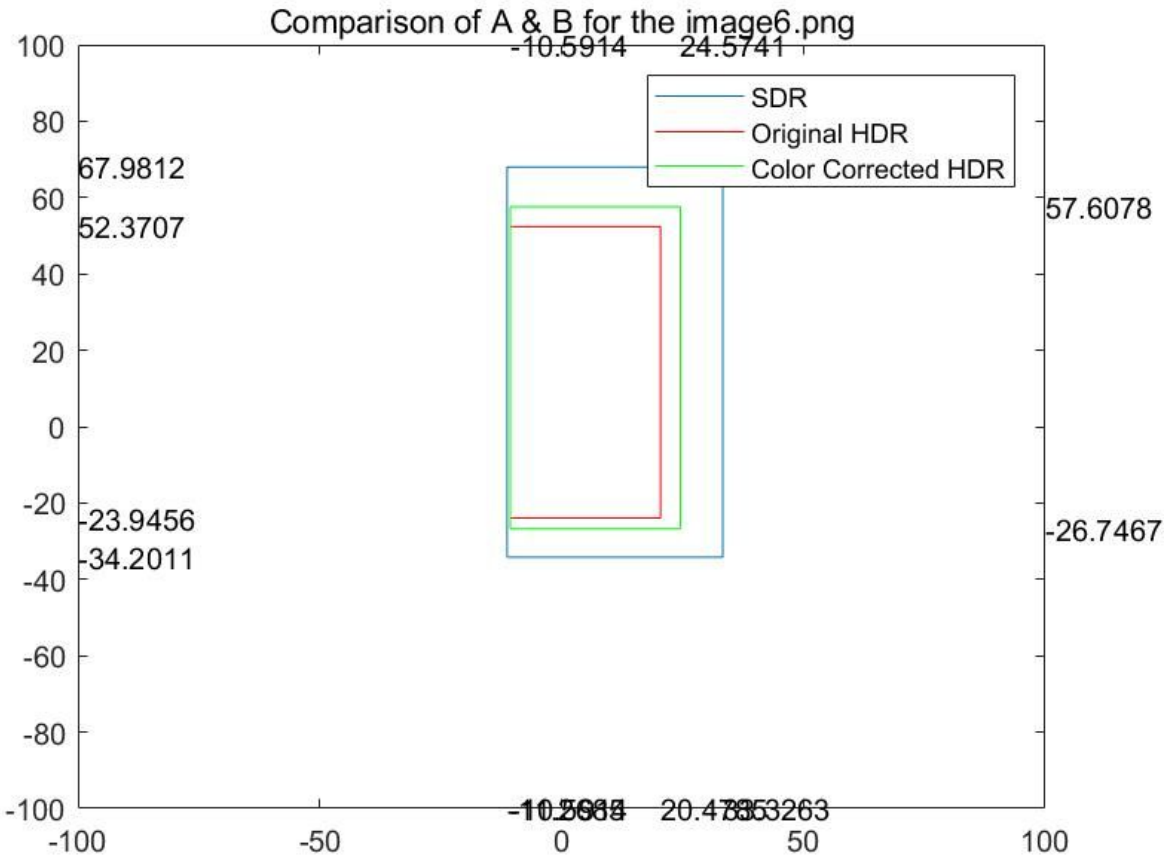


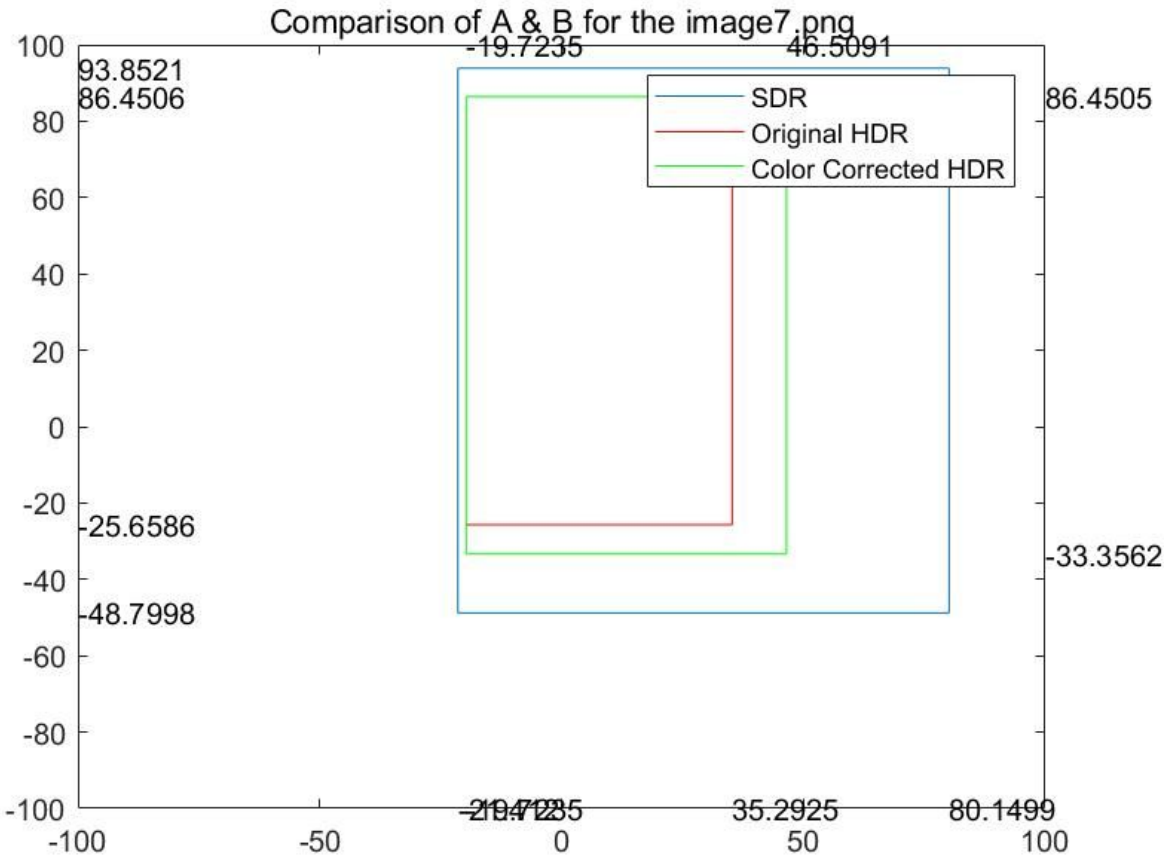

Comparison of A & B for the image2.png
SDR
Original HDR
Color Corrected HDR
-44.3664
45.7426
79.4723
72.4274
76.4155
-31.0914
-51.7545
-37.3097
35.1867
80.1116
100
80
60
40
20
0
-20
-40
-60
-80
-100
-100
-50
0
50
100


Comparison of A & B for the image3.png
SDR
Original HDR
Color Corrected HDR
-27.339
36.2881
79.2086
69.35
76.2849
-32.6054
-45.7408
-93.1681
26.1084
55.1464
100
80
60
40
20
0
-20
-40
-60
-80
-100
-100
-50
0
50
100


Comparison of A & B for the image4.png
SDR
Original HDR
Color Corrected HDR
-19.2596
33.2031
81.1736
66.8881
70.2596
-33.1227
-46.2041
-70.8415
23.0051
52.8869
100
80
60
40
20
0
-20
-40
-60
-80
-100
-100
-50
0
50
100


Comparison of A & B for the image5.png
SDR
Original HDR
Color Corrected HDR
94.3324
87.6658
-36.0068
39.729
87.6657
-29.8218
-39.1368
-76.2977
-48.8556517
29.0794
81.6624
100
80
60
40
20
0
-20
-40
-60
-80
-100
-100
-50
0
50
100


Comparison of A & B for the image6.png
-10.5914
24.5741
SDR
Original HDR
Color Corrected HDR
67.9812
52.3707
57.6078
-23.9456
-34.2011
-26.7467
100
80
60
40
20
0
-20
-40
-60
-80
-100
-100
-50
0
50
100


Comparison of A & B for the image7.png
SDR
Original HDR
Color Corrected HDR
93.8521
86.4506
-19.7235
46.5091
86.4505
-25.6586
-33.3562
-48.7998
35.2925
80.1499
100
80
60
40
20
0
-20
-40
-60
-80
-100
-100
-50
0
50
100