CS270 Homework 1

Deadline 2021/3/31 24:00

Notes:

30 points for question 1, 40 points for question 2, 30 points for question 3, a total of 100 points. The example results provided are just for reference. Please try to achieve the best performance as you can. Discussions are encouraged and plagiarism is strictly prohibited, source code should not be shared in any form. Please submit your homework (report in .pdf format and your codes) to the "gradescope" platform with both subject and file name in this format. (CS270+ID+name+hw1) example, CS270_2019123321_张三_hw1.

Question 1 Defogging and Fogging

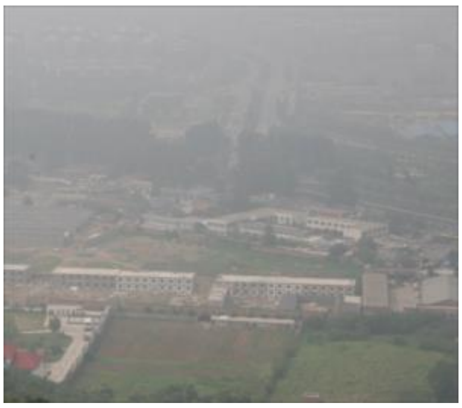
In this question, we first request that you can improve the quality of one given image using image enhancement methods implemented by yourself. For example, you could improve the image contrast by adjusting the histogram or try some image filters. As shown in Fig. 1, (a) shows the original image which looks foggy; (b) gives one example result which seems clearer after improving the contrast.

Then, we need you to impair the image quality of one normal image (see Fig. 2), which is contrary to the task above. You would make the clear image foggy after changing the contrast. By accomplishing these 2 tasks, you will understand better the process of image enhancement.

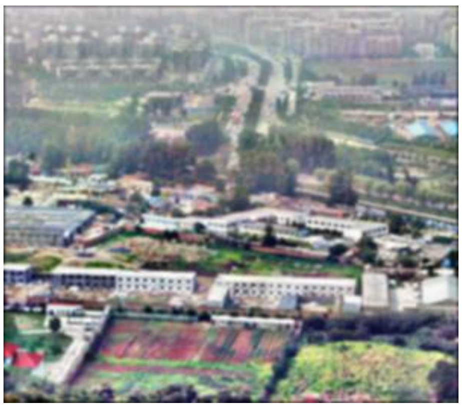
Tasks:

(1) Please describe your algorithms in words or flowcharts.

(2) Write your own codes to realize the requested processing and try not to use library functions.

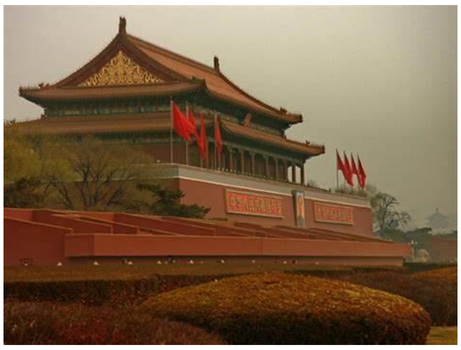


(a) Original image

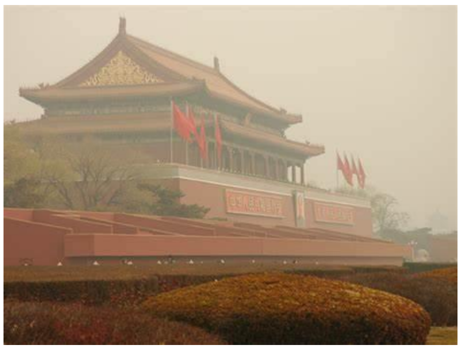


(b) Improved image

Fig 1: Examples of image enhancement.



(a) Original image



(b) Foggy image

Fig 2: Examples of image fogging.

Checkpoints

You need to answer these following questions in your report.

1. How do you implement your algorithm? Describe it by flow charts or words. (6 pts)
2. Describe implement details of your code. (6 pts)

You need to show these following figures in your report.

1. Results of defogging processing (like Fig. 1(b)). (9 pts)
2. Results of fogging processing (like Fig 2(b)). (9 pts)

Question 2 Image stitching

Image stitching is the process of combining multiple photographic images with overlapping fields of view to produce panoramic or high-quality views. Image stitching can be divided into four steps:

1. Detecting features, 2. feature matching, 3. image registration, 4. image blending.

In this question, we offer you three solutions to solve this problem:

1. Use the provided features to stitch the images directly. This can help you avoid steps 1 and 2. But please note that the features provided by the teaching assistant may be mismatched. Therefore, a direct stitching may not get a very satisfying result.

2. It is suggested to re-match the features (figure them out by yourself) provided by the teaching assistant to get some correctly matched features. Then, your results should look better.

3. Perform features extraction by yourself, match the extracted features, and perform image stitching based on these matched features.

Note: Please do not manually delete incorrect matches or add correct matches. If found (in your code), half of the points of this question will be deducted.

In step 3, choose your transformation method according to your needs. You can choose between linear transformation (Affine or Homography) and nonlinear transformation, and note that, different transformation methods bring different results.

Bonus: Image blending is also important in image stitching, as it was yet not introduced in the class and consider the amount of work to implement it, this part is a bonus but not mandatory. A good image stitching line should look "seamless". If you implement this function, we will add extra points (up to 10% of this question) to your homework based on your results

Sample data

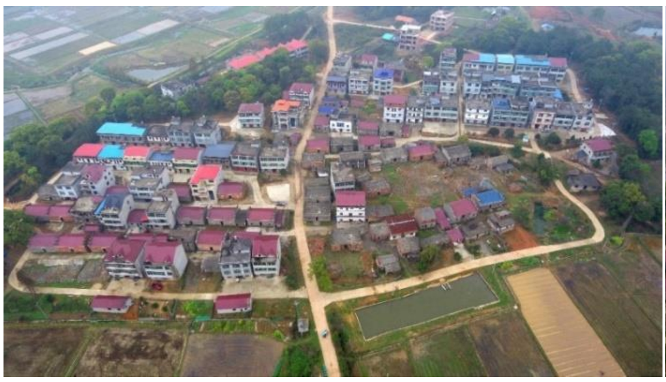


Fig 3

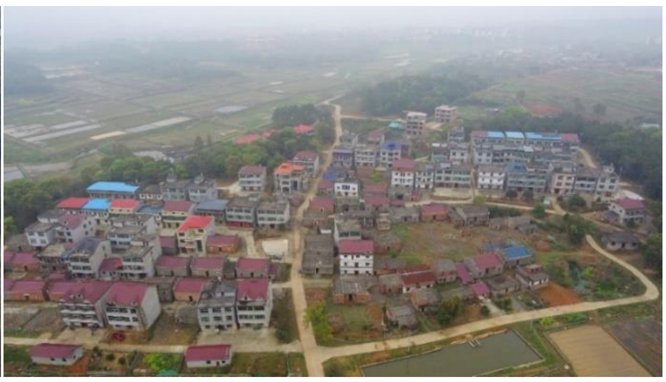


Fig 4

Fig. 3 and Fig. 4 are a group of photos of the countryside images. They are an example of the input images for this question. We will also provide you some matching points, as shown in Fig 5. Note that, for clarity, we only show a small amount of them. In fact, the number of points is far more greater than the number of points shown in Fig. 5, and there are many mismatches.

After linear transformation, we could get a registered picture of Fig. 4. Blend it with Fig. 3, to get the "overlay image" (see the example shown in Fig 6 and Fig 7). If you are not interested in obtaining the bonus, this could be a reference of your result as well.

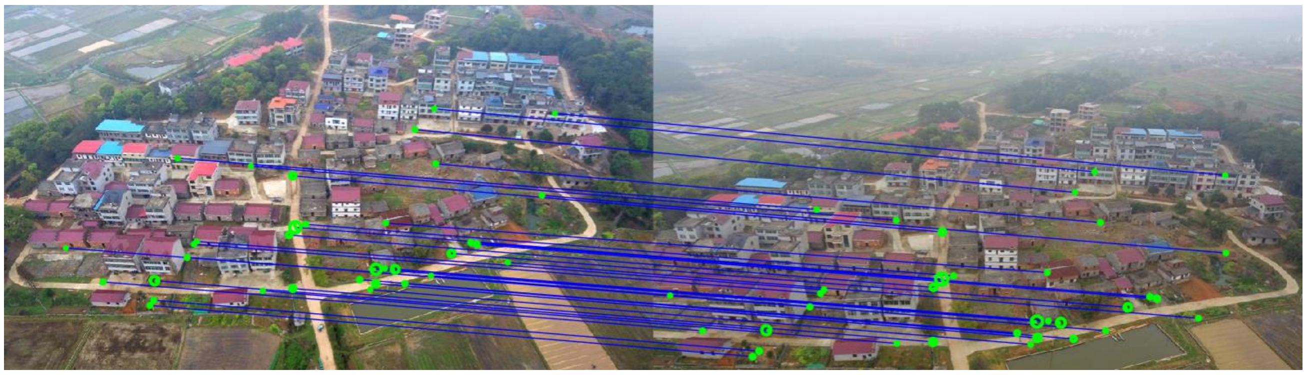


Fig 5

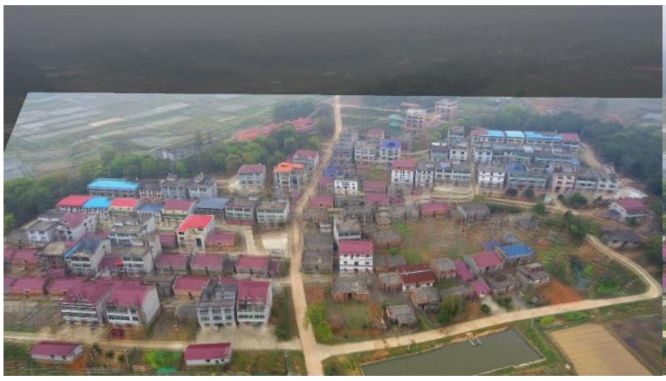


Fig 6

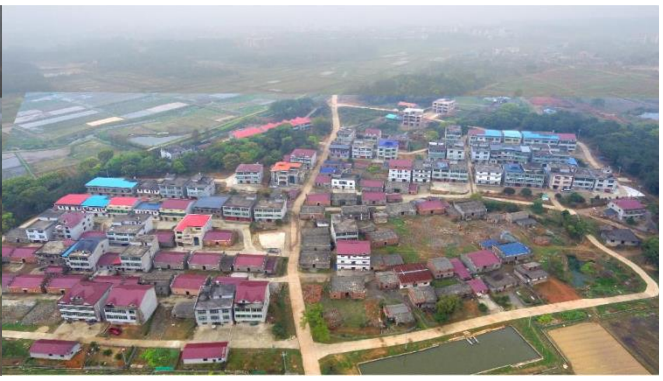


Fig 7

Hint for bonus: you may use "image stitching line" method.

Data interpretation

Each folder contains three files: two pictures, and a `.mat` file. You need to stitch two pictures together to complete the following checkpoints. You can load this mat file directly. This document contains three sets of variables, among which f1 is an array of 2*n, which represents the position of n key points in the first picture; f2 is an array of 2*m, which represents m key points in the second picture Position; match12 is an array of 2*c, representing c matching key points. For each row, the data in the first column represents the index of f1, and the data in the second column represents the index of f2.

Checkpoints

You need to answer these following questions in your report.

1. How do you extract features from the images? (3 pts)
2. How do you match the features? (3 pts)
3. How do you detect the mismatches? (6 pts)
4. Which transformation method do you use? How does it work? (8 pts)
5. (bonus) How do you find image stitching lines? (2 pts)

You need to show these following figures in your report.

1. Your matched features (like Fig 5). (4*2 pts)
2. Your matched result (like Fig 6 & Fig 7). (4*3pts)
3. (bonus) Your image stitching lines (like Fig 8). (4*0.5 pts)

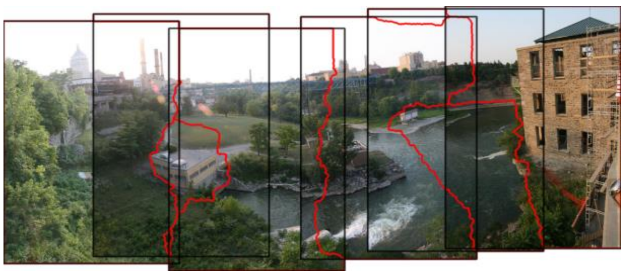


Fig 8

Question 3 Color Transfer Between Images

In this question, please implement color transfer between images using MATLAB or other languages. Here is a simple algorithm based on statistical analysis to impose one image's color characteristics on another. It chooses an orthogonal color space without correlations between axes to simplify the color modification process. Ruderman et al.'s perception-based color space $l\alpha\beta$ minimizes correlation between channels for many natural scenes. This space is based on data-driven human perception research which assumes that the human visual system is ideally suited for processing natural scenes. The mean and standard deviations along each of the three axes suffice to make an image take on another image's color characteristics. The color transfer result's quality depends on the similarity of the images in composition. Sometimes it fails and you can try to manually select two or more pairs of clusters in $l\alpha\beta$ space for transformation. Another possible extension would be to compute higher moments such as skew and kurtosis, which are respective measures of the lopsidedness of a distribution and of the thickness of a distribution's tails. Imposing such higher moments on a second image would shape its distribution of pixel values along each axis to resemble the corresponding distribution more closely in the first image.

Algorithm

(1) **RGB→$\boldsymbol{l\alpha\beta}$** Convert RGB signals to Ruderman et al.'s perception-based color space $l\alpha\beta$.

RGB→XYZ

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.5141 & 0.3239 & 0.1604 \\ 0.2651 & 0.6702 & 0.0641 \\ 0.0241 & 0.1228 & 0.8444 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

XYZ→LMS

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3897 & 0.6890 & -0.0787 \\ -0.2298 & 1.1834 & 0.0464 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

RGB→LMS

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3811 & 0.5783 & 0.0402 \\ 0.1967 & 0.7244 & 0.0782 \\ 0.0241 & 0.1288 & 0.8444 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Convert the data to logarithmic space to eliminate skew.

$$\begin{aligned} \boldsymbol{L} &= \log L \\ \boldsymbol{M} &= \log M \\ \boldsymbol{S} &= \log S \end{aligned}$$

LMS→$l\alpha\beta$

$$\begin{bmatrix} l \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & \frac{1}{\sqrt{6}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -2 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{L} \\ \mathbf{M} \\ \mathbf{S} \end{bmatrix}$$

(2) **Statistics and color correction**. Compute the means and standard deviations for each axis (3 in total) separately in $l\alpha\beta$ space. Subtract the mean from the data points

$$l^* = l - \langle l \rangle$$
$$\alpha^* = \alpha - \langle \alpha \rangle$$
$$\beta^* = \beta - \langle \beta \rangle$$

Scale the data points comprising the synthetic image by factors determined by the respective standard deviations:

$$l' = \frac{\sigma_t^l}{\sigma_s^l} l^*$$
$$\alpha' = \frac{\sigma_t^\alpha}{\sigma_s^\alpha} \alpha^*$$
$$\beta' = \frac{\sigma_t^\beta}{\sigma_s^\beta} \beta^*$$

Add the averages computed for the photograph

$$l'' = l' + \langle l_t \rangle$$
$$\alpha'' = \alpha' + \langle \alpha_t \rangle$$
$$\beta'' = \beta' + \langle \beta_t \rangle$$

(3) $\boldsymbol{l\alpha\beta}$**→RGB** Transfer results in $\boldsymbol{l\alpha\beta}$ to RGB to display it.

lαβ→LMS

$$\begin{bmatrix} \mathbf{L} \\ \mathbf{M} \\ \mathbf{S} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -2 & 0 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{6}}{6} & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} l'' \\ \alpha'' \\ \beta'' \end{bmatrix}$$

Raising the pixel values to the power of 10 to go back to linear space.

$$L = 10^{\mathbf{L}}$$

$$M = 10^{\mathbf{M}}$$

$$S = 10^{\mathbf{S}}$$

LMS→RGB

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 4.4679 & -3.5873 & 0.1193 \\ -1.2186 & 2.3809 & -0.1624 \\ 0.0497 & -0.2439 & 1.2045 \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix}$$

Task:

(1) The ‘houses.bmp’ is the source image and the ‘hats.bmp’ is the target image. Convert them to $l\alpha\beta$ color space. Use the ‘imshow()’ function to show the converted results.

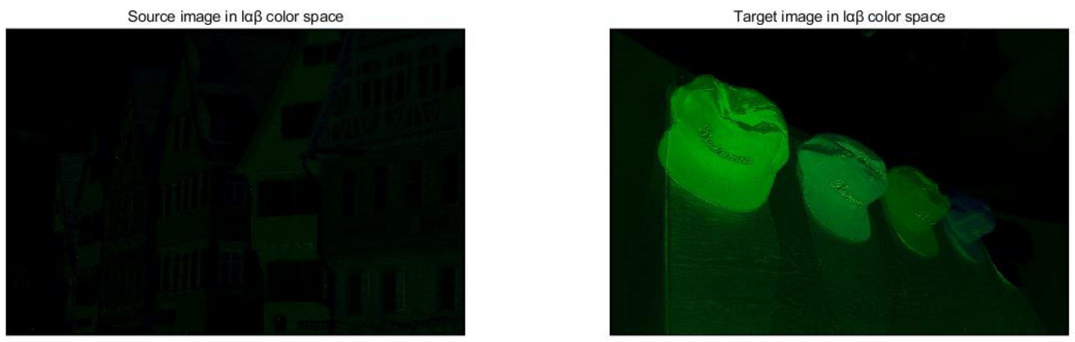

Source image in lαβ color space
Target image in lαβ color space

Fig 9: Reference results.

(2) Use the ‘imshow(,[])’ function to show the l, α, β three components of the ‘houses.bmp’ in $l\alpha\beta$ color space.

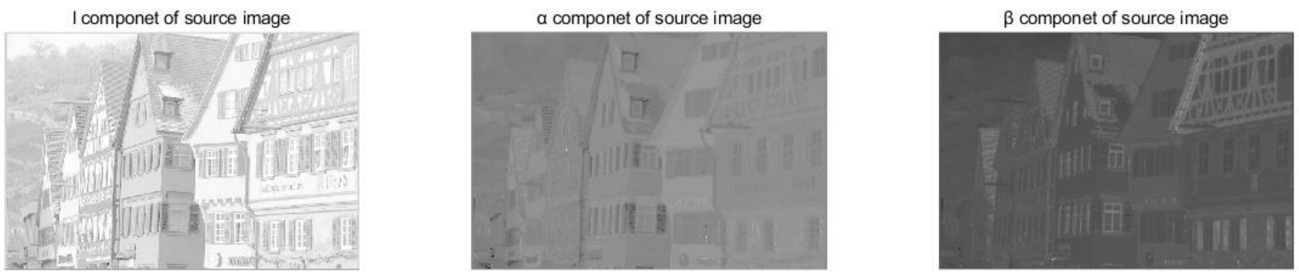

l componet of source image
α componet of source image
β componet of source image

Fig 10: Reference results.

(3) Implement the statics and color correction in $l\alpha\beta$ color space. Transfer the result back to the RGB color space. Use the ‘imshow()’ function to show the converted results in $l\alpha\beta$ color space and RGB color space.

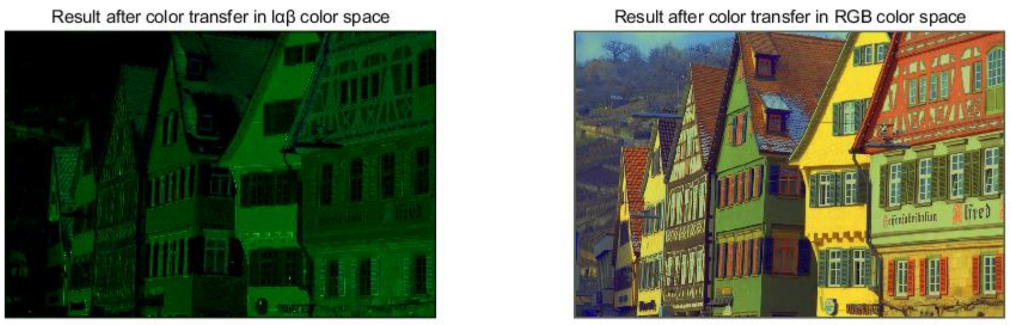

Result after color transfer in lαβ color space
Result after color transfer in RGB color space

Fig 11: Reference results.

(4) Repeat the above steps, transfer the ‘Moutains.png’ color to ‘Starry_night.png’ and transfer the ‘Starry_night.png’ color to ‘Mountains.png’. Show the transfer results.

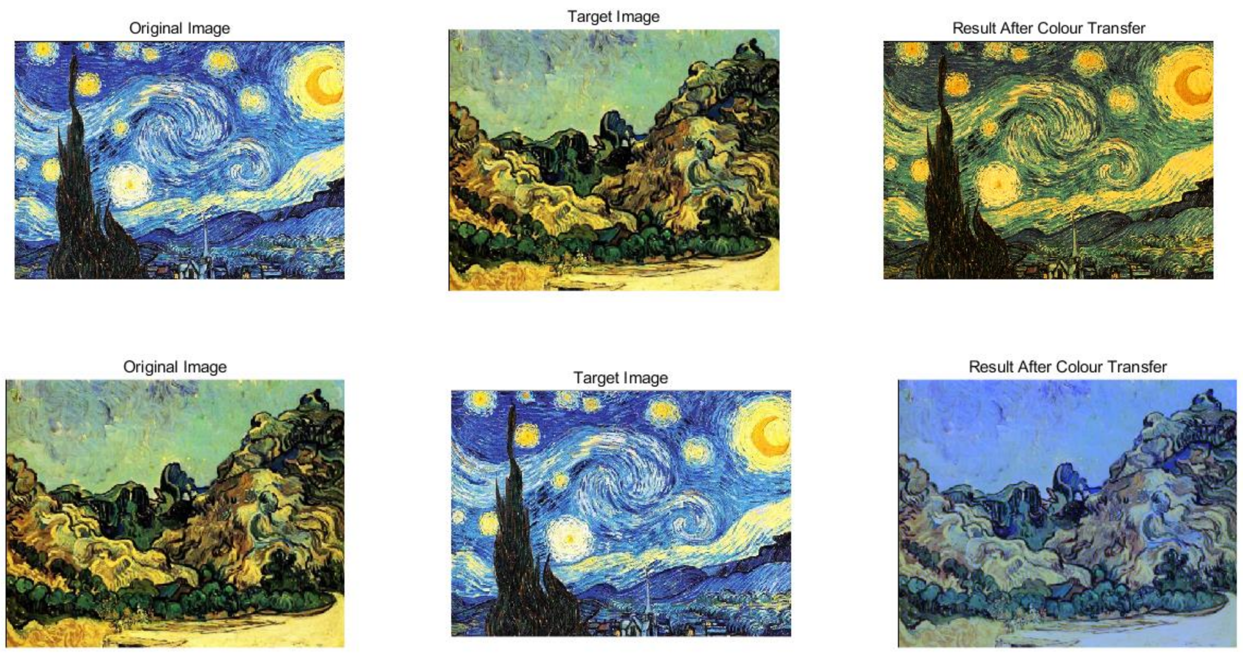


Fig 12: Reference results.

(5) Select two suitable images to implement color transfer and try to explain the transfer result.

(6) You can try to improve the algorithm to make your results better than the reference results.

Checkpoints:

You need to answer these following questions in your report.

1. Explain the transformation steps, and your coding implementation. (7 pts)

2. The improvements you made to the algorithm. (3 pts).

You need to show these following figures in your report.

1. LAB color space image (like Fig 9). (2*2 pts)
2. Component image of LAB (like Fig 10). (2*2 pts)
3. Result of your color transformation (like Fig 11). (2*2 pts)
4. Result of your color transformation (like Fig 12). (2*2 pts)
5. Your selected images and color transfer results (4 pts).

Tips:

(1) When performing the logarithmic operation, if the input value is 0, you will get -Inf. Add a very small number can avoid this situation.

References

[1] Reinhard E, Adhikhmin M, Gooch B, et al. Color transfer between images[J]. IEEE Computer graphics and applications, 2001, 21(5): 34-41.