Science and Technology Innovation 4I

color to gray scale

1. Introduction

For image processing in computers, sometimes we need to transform a color picture into a gray-scale one, which is called the decolorization. In fact, this can be achieved by a very simple method, that is, just take the linear luminance of the original picture and let it be the gray-scale value of the new picture. This is really simple and efficient, because the complexity is just O(n*m), and all that need to do is just to take a linear combination of the original pixel's different color channel, here is the description from wiki:

(R, G, and B, also in range [0,1]). Then, luminance is calculated as a weighted sum of the three linear-intensity values. The sRGB color space is defined in terms of the CIE 1931 linear luminance Y, which is given by

$$Y = 0.2126R + 0.7152G + 0.0722B.$$ [5]

In fact, decolorization can be viewed as a process of data compression, the original color image must have more info than the transformed gray image, that is, the decolorization must select and delete some info. Then, the question is what to delete and what to remain, which decides how we want the transformed gray-scale image to be. In fact, when we see an image, what remains in our brain is the contrast of the image, which is the key element of the image by which we can discriminate one thing from the environment or another thing. So, the key point of decolorization is to preserve the contrast of original image.

Here, we use the method of the article [Jung Gap kuk. Et al 2010][1], which is from ACCV2010. The method in this article is to solve a really large sparse linear equation and directly find the gray-scale value for every pixel, and both local and global contrast are considered.

The implementation use the m script of matlab, and the testing is performed in matlab 2011b and gnu octave 3.6.4, the testing data come from [Cadik,2008][2]. The results are all provided in this package and for most of the data, the result seems pretty ideal, but in fact there is one image that does not seem good which will be talked about later.

2. The conversion method

In fact, the core of conversion is the solving of linear system of equations, this evolves a really large matrix which can be 10000*10000 and above, but thankfully it is sparse and can be solved quite fast by matlab.

So, the main problem is to construct this linear system, and firstly we'll introduce the principle of the proposed method.

In fact, the whole thing is just to minimize an energy function, which encodes both of the local and global contrast, for the numerous color metrics, the method use Lab space metric explored in [Neumann, Et al 2007][3] to calculate the color space "distance":

$$\delta_{ij} = \sqrt[3]{(L_i - L_j)^3 + w_a(a_i - a_j)^3 + w_b(b_i - b_j)^3} \tag{1}$$

And for the local contrast, the method just use the standard 4-neighborhood system, that is up down left right, and for the global contrast, the method first pick some landmark pixels(the number is specified to 30) and then include the "distance" of all pixels in the image to these landmark pixels, the choosing method will be talked about later.

So, the final energy function will be:

With the above metric and landmark colors, the proposed energy is formulated as

$$f(\mathbf{g}) = (1-\lambda) \sum_{(i,j)\in E} (g_i - g_j - \delta_{ij})^2 + \lambda \sum_{i\in V} \sum_{k\in Q} (g_i - g_k - \delta_{ik})^2 \qquad (2)$$

g is the transformed gray-scale image as a vector of all the pixels, which is what we want to solve and get, E is the pixel pool of all the 4-neighborhood system pixel-pairs, V is all pixels in the image and Q is the selected landmark pixels of the image, and lambda is the ratio of local and global contrast. So, the **g** that minimized the energy value is the final answer, which in other words can preserve the contrast of the original image best.

To get the answer, we differentiate the energy function with respect to g_i and set it to zero, and finally what we got to do is to solve the huge linear system:

$$(1-\lambda) \sum_{j\in N_i} (g_i - g_j - \delta_{ij}) + \lambda \sum_{k\in Q} (g_i - g_k - \delta_{ik}) = 0, \quad i \in V \qquad (4)$$

where N_i is a set of neighbors of the ith pixel. Aggregating all the linear equations to make them in a vector form, we finally have the following linear system :

$$((1-\lambda)\mathbf{S} + \lambda M I + \lambda \mathbf{P})\mathbf{g} = \mathbf{b}, \qquad (5)$$

Solving this linear system is not so fast, because the matrix on the left is not symmetric, although we don't know how matlab solves it, but with the fast version described next which use an approximate way with a symmetric matrix, it really gets faster.

3. A fast approximate way

The ideal is to get a approximate solution which simplifies the solving of the linear system(at least in the matlab's point of view).

The main change is the assumption that the gray-scale difference between a pixel and the landmark pixel can be approximated to the color distance.

$$\sum_{i\in V} \sum_{k\in Q} (g_i - g_k - \delta_{ik})^2 = \sum_{i\in V} \sum_{k\in Q} (g_i - g_n + g_n - g_k - \delta_{ik})^2 \qquad (6)$$

$$= \sum_{i\in V} \sum_{k\in Q} \frac{1}{|N_i|} \sum_{j\in N_i} (g_i - g_j + g_j - g_k - \delta_{ik})^2 \qquad (7)$$

$$\sum_{i\in V} \sum_{k\in Q} \frac{1}{|N_i|} \sum_{j\in N_i} (g_i - g_j - (\delta_{ij}^k))^2 \qquad (8)$$

where $\delta_{ij}^k = \delta_{ik} + \delta_{kj}$.

And finally after some tough maths, the final linear system becomes symmetric:

$$\sum_{j \in N_i} (g_i - g_j - \delta'_{ij}) = 0, \ \ i \in V \tag{11}$$

where $\delta'_{ij} = ((1-\lambda)\delta_{ij} + \frac{\lambda}{2}\sum_{k \in Q}\delta_{ij}^k)/((1-\lambda) + \frac{\lambda|Q|}{2})$. The main difference

4. The local contrast

The local contrast part of the energy function is quite simple, it just calculate all the "distance" of the 4-neighborhood system pixel pairs, but here is a minor problem that the pixel in the corner does not have 4 neighborhood, that is you can't count the pixel at the far-away other side as the neighborhood, so for the pixel at the edge and at the corner, some special care must be taken to construct the **S** matrix. But, overall this part is still simple compared to the global part.

5. The global contrast

The difficult part is to construct the global contrast of the energy function, and the main task is to get the landmark pixels of the image.

The selecting is based on the so-called popularity method, the method clips each color sample to 3N bits(N bits for each channel), so the image will generate a histogram with 2^(3N) bins and we choose the most M bins according to how many pixels each bin has. By averaging the color of the selected bins, we get the popular colors, then search the image for the closest pixel for each color --- which will generate the landmark pixels.

This seems like a simple method, but we need to think more about it, in fact, the landmark pixels are the representatives of the image, that is, if using only those colors, the image can be reconstructed just like the original image.

According to the article, the parameters for the method are 3 bits and 30 landmarks, many images can be represented fine, for example, many of the testing images do fine for the method, that is they include the main colors in the image, the reconstructing of some images are shown here:

And for many of the testings, the reconstruct image and the original one are just the same, this is the case in which the original image has few colors, but sometimes the result is not so ideal and some of the important colors are missing, this is can happen when the number for that color is small in the image but it itself is so standing-out that it can not be ignored:

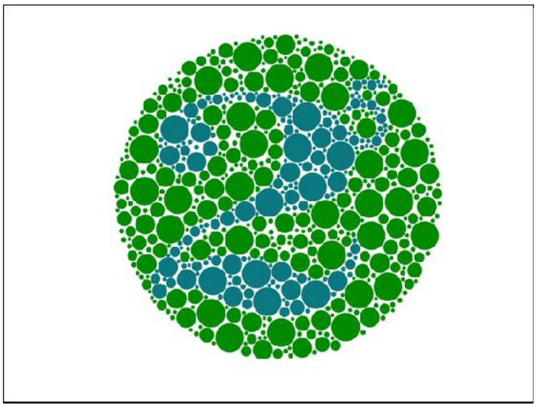
In this image, the sun is missing, because it just stand a really small space, but thankfully the red color of the image is not missing so the final result is not so bad. But imaging if the image of the sun in the water does not exist, maybe the red color will be missed, which might harm the final conversion.

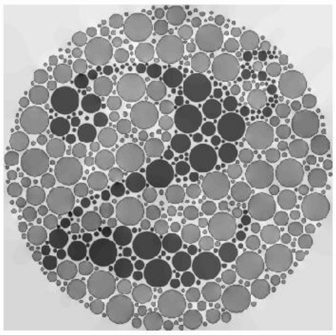
Nevertheless, the popularity method performed ok for the selecting of the landmarks for the global contrast, and one last thing is that if the original image is so colorful that there are so many colors in it, the method may become meaningless.

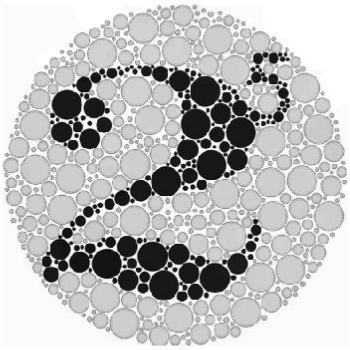
6. The lambda parameter

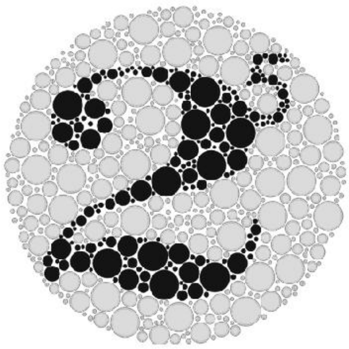
Remember in the energy function, there is a parameter lambda which adjust the ratio of the local contrast and global contrast, in fact, when lambda equals to 1, only global contrast is considered and if it is set to 0, only local contrast will be considered.

The conversion of the following image mostly shows the impact of lambda, which is also showed in the article, the four images are the original, lambda = 0,0.3,0.8:









However, for the some other images, the difference may be not so obvious, maybe this is because this image is quite simple and consists mainly two colors, so the global contrast is just the contrast of blue and green, so the adjustment of lambda can greatly influence the result.

7. About the color space and Problems of the method

The article use the Lab color space to calculate the "distance of two colors", and this method is discussed in [Neumann, Et al 2007][3], however there are these two parameters wa and wb we are not quite sure, here is what it said:

> the max feature, but it is also near to the square-root. Let $A = w_a * a$ and $B = w_b * a$, where w_a and w_b in interval [0.2..0.6] are weight factors to reduce the chrominance-luminance ratio. The equivalent luminance has to be smaller than a CIE color difference value, which can go over 200. Our new formula is as follows:
>
> $$\Delta = (\Delta L^3 + \Delta A^3 + \Delta B^3)^{1/3}. \qquad (1)$$

In the testing, we just set them to 0.2, but this lead to two weired example, it is the fruit.png and 155_5572_jpg.png image whose result is some kind of weired:

In fact, at first, we use the rgb color space to calculate the “distance”, and the result seems not so strange, however the rgb way have severe problems that for many cases they may not performing well, and thankfully for all other testing pictures, the Lab color space way does not generate such kind of strange results.

8. About the slow and fast method

The original method proposed in the article concerns a non-symmetric matrix, and then it show an approximate method that change it ti a symmetric matrix to speed up, and in fact, the speeding up really make sense. Although, we know nothing about the way that linear system is solved, that is all given to matlab or octave, but the comparison really show the differences. The testing data includes those two sets and for more information, just check the log for the testings.

9. The testing

The final testing is performed in window 7 environment and matlab 2011b, you can see two sets of results which are the fast and slow method. In fact, the scripts also are tested in the environment of Linux and gnu octave which is mainly the testing when writing and debugging the scripts. The testing data comes from [Cadik,2008][2], except for the two strange results mentioned before, other results seem reasonable.And here is some examples:

References:

[1] Jung Gap Kuk, Jae Hyun Ahn, Nam Ik Cho : A Color to Grayscale Conversion Considering Local and Global Contrast. ACCV 2010, Part IV, LNCS 6495, pp. 513-524

[2] Cadik, M: Perceptual evaluation of color-to-grayscale image conversions. Pacific Graphics 27, 1745-1754 (2008)

[3] Neumann L., Cadik, M., Nemcsics, A.: An efficient perception-based adaptive color to gray transformation. In: Computational Aesthetics in Graphics, Visualization, and Imaging, pp. 73-80 (2007)