Monochrome photographs colorization

Yinheng Chen u6341895@anu.edu.au
Shidong Pan u6342277@anu.edu.au
Zhibo Zhang u6015337@anu.edu.au

Abstract

Adding colors to grayscale image by computer-assisted is defined as colorization.To solve the problem, we used the algorithm which was proposed by [3]. The main idea of the algorithm is that two neighboring pixels should have similar color if their luminances are similar and it also contains a optimization method. Some processing methods are also necessary. The experiments and results section represent the colorization results by implementing the algorithm with different methods.

1 Introduction

Adding colors to monochrome photos and black-and-white films by computer-assisted is defined as colorization and it has been a heat topic since the early 1970s. When colorization just was introduced into movie industry, limited by the poor performance of computer software and algorithms at that time, it is particularly difficult for objects segmenting, and even hair and face regions cannot be correctly separated. Generally, this technology includes two main parts: segmenting pictures and adding colors to segmented regions. However, it is difficult to segment picture at some fuzzy or complex boundaries, besides, for some nongrid regions, identify target region is difficult. Levin et al[3].

In this project, we intend to implement the monochrome photographs colorization algorithm based on the recommend paper, basically applying edge detection based on YUV channels of the image and coloring those sections depending on their intensity. Furthermore, we did some experiments and evaluations on tuning parameters, extracting and painting the scribble colors in different ways. Also, a simple user interface is designed to illustrate outputs.

2 Related work

Colorization is an active and challenging area of computer vision research and it attracts a lot of interest in the image and film editing community. Some approach has produced impressive colorizations performance only using scribbled colors provided as a small amount of user input to produce a completely colorized image.

Yatziv et al.[8], proposed a computationally simple, yet effective, approach of colorization. The method is mainly based on the concepts of “luminance weighted chrominance blending” and “fast intrinsic distance computations” to obtain the high quality colorization results for both images and videos after the user provides a reduced set of chrominance scribbles.

Another approach is introduced by Sapiro [5], Firstly, given by its gradient information and based on the geometry and structure of the monochrome luminance input, the geometry and structure of the image will be obtained. By the color scribbles provided by user or side information and the gradient information brought in by the monochrome data, solving a partial differential equation can obtain the color which is expected.

Compared to previous approach, Zhang et al.[9], implemented a system which is based on feed forward pass in a CNN to complete the automatically colorization task. They apply "class rebalancing at training time to increase the diversity of colors" and train their CNN on over a million color images.

Same as Zhang et al.'s method, Cheng et al.[1], also aim at using neural net work to achieve high quality fully automatic colorization. "An extremely large scale" reference database (that contains sufficient color images) is used to obtain the most reliable solution to the colorization problem.

3 Algorithms and methods

3.1 Acknowledgement

We use the algorithm introduced in [3]. However, we also implemented extra necessary methods to achieve and visualize the results of colorization. In our implmentation, we worked on the YUV color space. In YUV, Y is the luminance channel while U and V are the chrominanc channels [2].

3.2 Image preprocessing

This step is quite simple. In our implmentation, there are many types of image. To simply the computing, we first convert images to 8-bit images in RGB color space, and then we convert these images to YUV channels. The equation of YUV channels to RGB channels is given below:

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ U &= C_r = R - Y \\ V &= C_b = B - Y \end{aligned} \tag{1}$$

3.3 Image hinting

There are many ways that we can draw hints on our image. We can use existed software to paint color scirbbles on our image, or achieving this goal by coding. Another aspect that we need to consider is which colors are appropriate. We can use a palette to get any color we want, or we can extract colors from other color image. Results of these methods will be discussed in our experiments part.

3.4 Algorithm

fter the previous two steps, we get one grayscale image and one hinted image. Images are decomposed to YUV channels. We combine the Y channel of the grayscale image and UV channels of hinted image to a new matrix, since hint image will change the Y channel of the grayscale image and we wish to ensure that two neighboring pixels **r,s** should have similar colors if their luminances are similar. Two pixels are neighbors if their image location are nearby. To minimize the difference between color $U(r)$ at pixel r and the weighted average of the U colors at neighboring pixels, we have:

$$J(U) = \sum_r \left(U(r) - \sum_{s \in N(r)} w_{rs} U(s) \right)^2 \tag{2}$$

w_{rs} is a weighting functions that sums to one, for two neighboring pixels r,s , w_{rs} is large $Y(r)$ is similar to $Y(s)$ while small if $Y(r)$ is similar to $Y(s)$ are significantly different. There are two weighting functions that we have used, one is based on the squared difference two intensities:

$$w_{rs} \propto e^{-(Y(r)-Y(s))^2/2\sigma_r^2} \tag{3}$$

The second weighting function is based on the normalied correlation between two intensities:

$$w_{rs} \propto 1 + \frac{1}{\sigma_r^2} \left(Y(r) - \mu_r \right) \left(Y(s) - \mu_r \right) \tag{4}$$

We need to colorize image in a series of successive frames. Between two successive steps(frames), after accounting for motion, two pixels are neighbors if their locations are nearby. In our implmentation,

window size w is defined, where pixels in a square with size w should be considered as neighbors in each frame. So, two pixel (x_0, y_0, t) and $(x_1, y_1, t+1)$ are neighbors if:

$$||(x_0 + v_x(x_0), y_0 + v_y(y_0)) - (x_1, x_2)|| < w \tag{5}$$

where $v_x(x_0)$ and $v_y(y_0)$ can be computed by using a standard motion estimation algorithm [4].

These weighting functions are usually referred to as affinity function[6][7]. We have experimented both two affinity functions and they have noticeably differenct results. Once we get a combined image, we are given a set of pixels where their colors are specified by hinting, and then we minimize $J(U)$,$J(V)$ to these constraints. Since the constraints are linear, to solve this problem is to solve a large spare system of linear equations. There are a number of standard methods to solve these linear equations. In our implmentation, we use $spsolve$ function in Python scipy.sparse.linalg module.

4 Experiments and results

One problem for existed softwares is that some softwares will change the format and luminance of the whole grayscale image after drawing the hint; therefore, this method is discarded in our experiments. Instead, we design a simple interactive platform by using Opencv package to drawing hint on our images.

By implementing the algorithm described before, different images with different parameters were tested including different weight functions, different neighbour window size. For those large images, only small window size were tested due to limited running time. The results images and running time were recorded. Three images (figure 1) and two weight functions were used to test which are shown below. The hinted images are shown in figure 2, and we choose colors from modern pictures (9).

(a) Audrey

(b) Greek

(c) Chinese workers

Figure 1: Three test images

(a) Audrey

(b) Greek

(c) Chinese workers

Figure 2: Three test images

For weighting function (4), the first and second image in Figure 1 were tested. Different window size 1,3,5,7 were used. For Third image, to control running time, only size 1 and 3 were tested. The results are shown below in figure 3, figure 4 and figure 5.

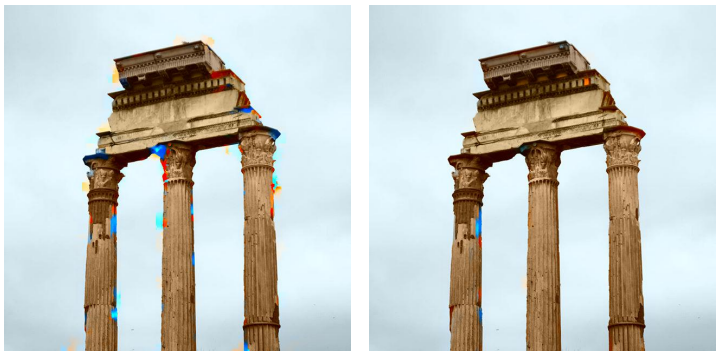
(a) window size 1

(b) window size 3

(c) window size 5

(d) window size 7

Figure 3: Audrey test image with window size 1,3,5,7 by using weighting function (4)



(a) window size 1 (b) window size 3

(c) window size 5 (d) window size 7

Figure 4: Greek test image with window size 1,3,5,7 by using weighting function (4)

(a) window size 1 (b) window size 3

Figure 5: People test image with window size 1,3 by using weighting function (4)

From the results from weighting function (4), it can be found that when window size is small the running time is small as well, but the result is unpleasant. With increasing window size, the quality of result becomes better, but the running time multiply grows. Besides, the running time in table 1 is based on high quality hint on images as being shown below, where w represents window size and t represents running time.

Table 1: Running time by using weighting function (4)

$t(s)$ / w \ Image	Audrey	Greek	People
1	8	59	89
3	29	403	2789
5	191	4579	NaN
7	336	15567	NaN

To test weight function (3), only Audrey image will be used to test different window size. The other two images will also be tested by using windown size 1. The results are shown in figure 6.

By testing different window size of weighting function (3), from the result in Figure 6, there is no much differences between different window sizes. Also, the result of colorization by using weighting function (3) is much better than that by using weighting function (4).

(a) window size 1

(b) window size 3

(c) window size 5

(d) window size 7

(e) Greek

(f) Chinese workers

Figure 6: Test images with window size 1,3,5,7 by using weighting function (3)

Also, the quality of hint is directly effected the colorization quality. Ways were tested, one way different from previous printing way was shown below. This hinting way is much rough. With this different hinting way, different window size were tested. The results of Audrey image is shown in figure 7 and figure 8.

The results of Audrey image by using rough hinting way are almost same as those by using fine hint. One significant difference is the lip because we did not paint color on the lips in the image when we roughly drew hint.

We have also tested other images (Greek and People) by using the rough hinting way. Since Greek image has less features, the colorization result of Greek image is similar to that of Audrey image. As for the Chinese worker image, there are lots of small areas that need to be painted in different colors; therefore, using an rough hint will greatly decrease the colorization quality.

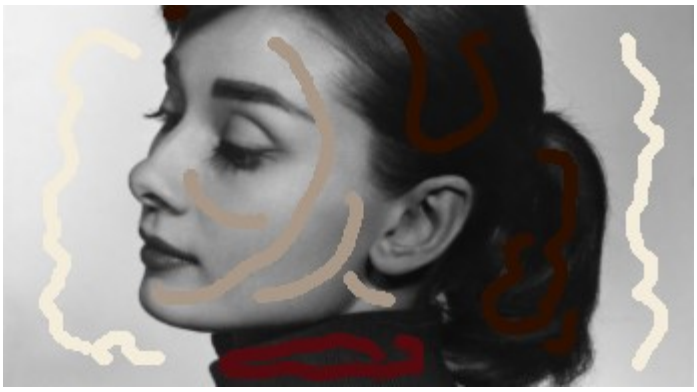


Figure 7: Audrey hinted image (rough)

(a) window size 1

(b) window size 3

(c) window size 5

(d) window size 7

Figure 8: Audrey test image with window size 1,3,5,7 by using weighting function (3)

5 Conclusion

Colorization has a wide application in the computer vision field and it will bring new life to the monochrome photographs or films. In our project, we used an optimization method to implement a colorization model based on intensity and edges detection, by solving a number of linear equations. However, there are many factors that impact the quality of colorization outcome: weighting function in finding neighbour pixels, windows size, image size and the specific level of painting hint as color scribble. After experiments, we found that combining the weighting function (3) and setting window size as 3 will efficiently have a expected colorization result.

As for making the monochrome images look more antique, excepting applying appropriate parameters mentioned above, extracting softer color (less saturated and edgy) colors from modern photos is another important aspect.

Compared to other existing colorization approaches that based on neural network method, our model has a much faster speed and does not rely on a large-scale dataset. Meanwhile,the color extraction will keep the flavor of historical photoraphs in a yellowish tone.

6 Future work

Currently, if the size of the monochrome image is large, the program will be too sluggish to get outcome. To improve the efficiency, concurrent algorithm can be introduced into our model to solve it.

Nevertheless, sophisticated images are requied to spend more time on manually painting scribbles as hints. For this problem, although there is no significant method to improve it, we can define a better weighting function to decrease the workload of manual painting as much as possible.

We are also expected to design and create a much beautiful and friendly interactive system which contain more functions, such as selecting colors from mutiple images, represent mutiple results and etc.

7 Summary of learning outcome

Through the process of conducting the project step by step, first of all, we successfully applied the discrete knowledge learnt from course on solving a real and completed problem. For example, the selection of YUV channel deepened our understanding of image definition and edge detection by intensity also inspired us more thought about edge detection algorithms. Secondly, we gain valuable experience about how to complete a big project. Intensive content can be broken down into pieces like algorithm, evaluation, experiments, etc. Therefore, it allows us to tackle this larger and more complex problems. Also, communication and teamwork skills are developed in the semester-long period and a shared identity with other group members was established. We kept regular meetings weekly and shared thoughts with others once someone had some breakthroughs. In addition, the project broadens our horizon, makes us recog-nize the state-of-the-art of different Computer Vision research directions, increases the capacity to analyse and evaluate research works on this field.

References

[1] Cheng, Z., Yang, Q. and Sheng, B., 2015. Deep colorization. In Proceedings of the IEEE International Conference on Computer Vision (pp. 415-423).

[2] Jack, K. (2007). Video demystified. Amsterdam: Newnes/Elsevier.

[3] Levin, A. et al. (2004) 'Colorization using optimization', in ACM SIGGRAPH 2004 Papers on SIGGRAPH '04. New York, New York, USA: ACM Press, p. 689. doi: 10.1145/1186562.1015780.

[4] Lucas, B.D. and Kanade, T., 1981. An iterative image registration technique with an application to stereo vision.

[5] Sapiro, G., 2005, September. Inpainting the colors. In IEEE International Conference on Image Processing 2005 (Vol. 2, pp. II-698). IEEE.

[6] Shi, J. and Malik, J., 2000. Normalized cuts and image segmentation. Departmental Papers (CIS), p.107.

[7] Weiss, Y., 1999. Segmentation using eigenvectors: a unifying view. Proceedings of the Seventh IEEE International Conference on Computer Vision.

[8] Yatziv, L. and Sapiro, G., 2006. Fast image and video colorization using chrominance blending. IEEE transactions on image processing, 15(5), pp.1120-1129.

[9] Zhang, R., Isola, P. and Efros, A.A., 2016, October. Colorful image colorization. In European conference on computer vision (pp. 649-666). Springer, Cham.

Appendix 1: Peer review

Table 2: work load distribution and individual contribution

Content / Group Member	Work distribution	Contribution ratio
Yinheng Chen(u6341895)	code implementation, algorithm improvement, colorization experiment, report formatting and writing.	33.3%
Shidong Pan(u6342277)	results collecting, colorization experiment, presentation preparation, report formatting and writing.	33.3%
Zhibo Zhang(u6015337)	code implementation, algorithm improvement, colorization experiment, report writing.	33.3%

Appendix 2: Image acknowledgement

Links of our test images and color images:
Chinese workers: https://www.pinterest.com.au/pin/791366965741893636/
Chinese soldiers: http://onlineresize.club/pictures-club.html
Audrey color: https://www.anews.com/us/n90273367-see-audrey-hepburn-in-gorgeous-rarely-seen-photos-in-honor-of-her-89th-birthday/
Audrey grey: https://www.fresha.com/bombshell-beauty-and-brow-studio-fgait33w
Greek: https://rdougwicker.com/2018/05/18fun-photo-friday-54-days-at-sea-rome-favorites-black-white-edition/

(a) Audrey

(b) Greek

(c) Chinese soldiers

Figure 9: Three color images

Appendix 3: Interactive interface

For this interactive interface, we get color from right image (status is 0) and hint color (status is 1) on left grayscale image.

Figure 10: Interactive interface