Color Image Processing

Digital Image Processing

by Trần Minh Hiếu, Nguyễn Gia Phong, Nguyễn Văn Tùng,
Nguyễn An Thiết and Nguyễn Thành Vinh

July 6, 2020

Contents

1 Introduction **2**
1.1 Brief Description 2
1.2 Authors and Credits 2

2 Color Spaces **3**

3 Color Image Enhancements **3**

4 Pseudo Color Rendering **3**

5 Conclusion **4**

6 References **4**

1 Introduction

1.1 Brief Description

Color images existed long before the rise of computing and digital image processing. While most techniques of monochrome image processing such as blur and edge detection can be directly applied to color images, others require modification. Furthermore, there exists procedures specific only to color images. In this project, we try to implement some of these techniques and note down our findings.

1.2 Authors and Credits

The work has been undertaken by group number 8, whose members are listed in the following table.

Full name	Student ID
Trần Minh Hiếu	BI9-101
Nguyễn Gia Phong	BI9-184
Nguyễn Văn Tùng	BI9-229
Nguyễn An Thiết	BI8-174
Nguyễn Thành Vinh	BI8-187

We would like to express our special thanks to Dr. Nghiêm Thị Phương, whose lectures gave us basic understanding on the key principles of digital image processing. The color image processing lecture notes from the UMSL's CS 5420 course [1] also help us gain initial intuition on the matter.

*`https://github.com/McSinyx/recipe`

2 Color Spaces

3 Color Image Enhancements

4 Pseudo Color Rendering

By mapping each intensity level to a color, one may derive a pseudo color image from a greyscale images. Typical usage of such technique is in thermal imaging, elevation and medical imaging to help the human visual system pick out detail, estimate quantitative values, and notice patterns in data in a more intuitive fashion [2].

In OpenCV, this is available under `applyColorMap`. It is also trival to reimplement this function only using NumPy:

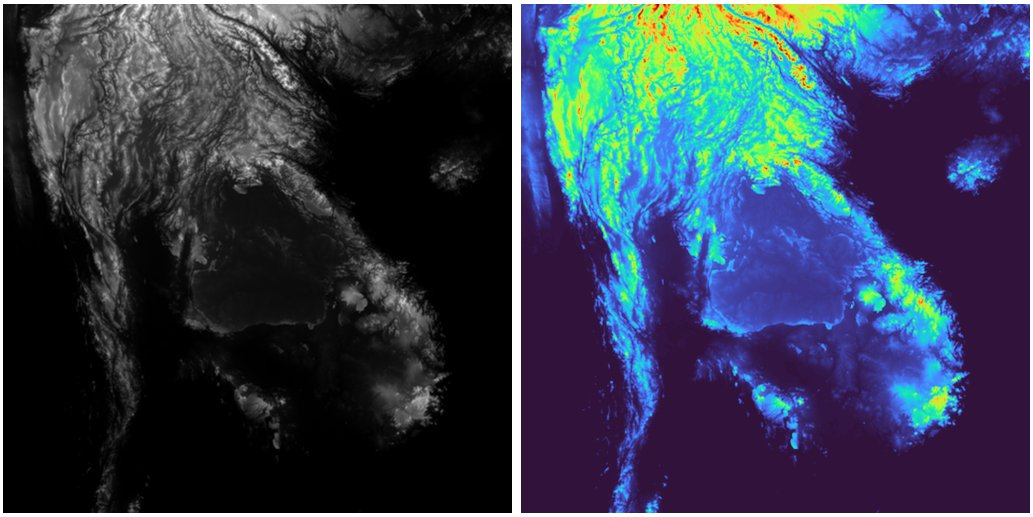
```
import numpy as np

def map_color(grey, mapping):
    r = np.vectorize(lambda i: mapping[i][0])
    g = np.vectorize(lambda i: mapping[i][1])
    b = np.vectorize(lambda i: mapping[i][2])
    # OpenCV uses BGR by default for whatever reason.
    return np.stack((b(grey), g(grey), r(grey)), axis=-1)
```

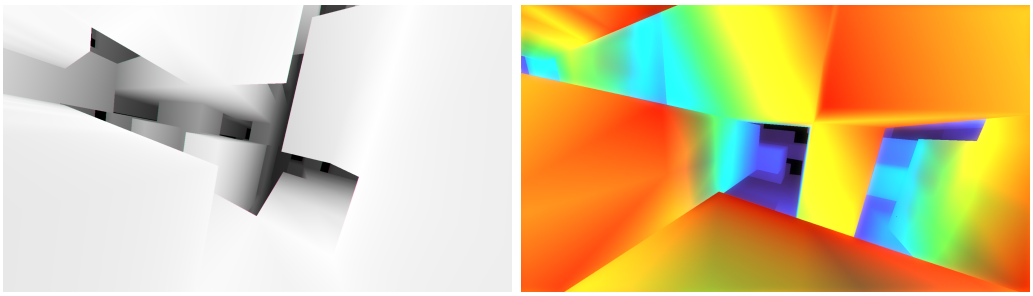
For demonstration, we are going to use the Turbo colormap [2]. We initially considered solely changing the hue based on intensity, which is also known as the rainbow map, however it is not a visually intuitive mapping [3].

Firstly, we tried to apply the mapping on the heightmap of mainland Vietnam and its neighboring regions[†]. As seen in the side-by-side comparison, the colormapped image allows the human eyes to notice more details, especially the high mountains in the north and the Khorat Plateau (center of the image).

[†]The original images are taken from heightmapper: `https://tangrams.github.io/heightmapper/`



In addition, we experimented with pseudo lighting, that is, use colormaps in place of normal greyscale lighting. The experiment was carried out on Phong's game Axuy, where colormapping proved to be an enhancement on helping players detecting shooting range (Axuy is a first person shooter game). The video where the game is in action is available on YouTube[‡].



5 Conclusion

6 References

[1] Sanjiv K. Bhatia. "Color Image Processing". *CS 5420: Digital Image Processing*. University Of Missouri—St. Louis, Fall 2018.

[2] Anton Mikhailov. *Turbo, An Improved Rainbow Colormap for Visualization*. Google AI Blog, August 20, 2019.

[‡] `https://www.youtube.com/watch?v=QVGAaoordpk`

[3] Noeska Smit. *Rainbow Colormaps—What are they good for? Absolutely nothing!* medvis.org, August 21, 2012.