



Due Date: 23:59 pm on Thursday, November 3rd, 2022

Images of the Russian Empire: Colorizing the Prokudin-Gorskii Photo Collection¹

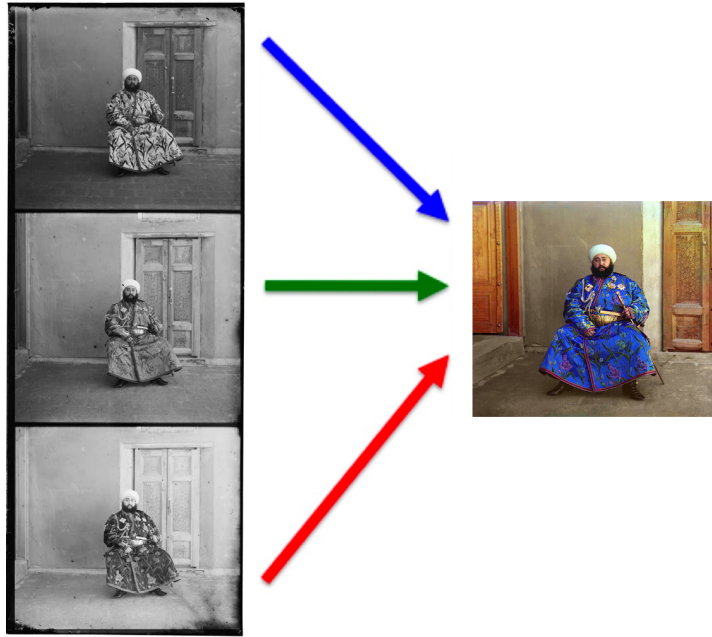


Figure 1: Blue, Green, Red negatives to RGB image.

Background

Sergei Mikhailovich Prokudin-Gorskii (1863-1944) [1], a Russian chemist and photographer, is considered as a pioneer in color photography. With Tsar's special permission and funding, he traveled across the Russian Empire, and took color photographs of people, daily life, buildings, landscapes, etc., documenting the early 20th-century Russia. He produced these color photographs by using a simple but ingenious idea: recording three exposures of every scene onto a glass plate using a blue, a green, and a red filter. These negative plates are now in possession by the Library of Congress of U.S.A., and have been recently digitized. They are available on-line at [2].

Overview

The goal of this assignment is to automatically produce color images from the digitized Prokudin-Gorskii glass plate images by applying some simple image processing techniques. For that, you will need to extract the red/green/blue color channels from the digitized plate images, and find the best alignment so that they form an RGB color image with few visual artifacts. You can download a starter code and some sample plate images from the course webpage.

Details

Your program will take a glass plate image as input and produce a related color image as output. It should have the following structure:

¹Credits: This assignment is a simplified version of the one that Alexei A. Efros developed for his Computational Photography class.

1. Divide the input image into three equal parts corresponding to RGB channels.
2. Align the second and the third parts (G and R channels) to the first one (B channel).

The simplest strategy to align the parts is to perform an exhaustive search over a window of possible displacements (e.g. [-15,15] pixels), and take the displacement that gives the best matching score. Some possible matching metrics are as follows:

- Sum of squared differences (SSD)
- Normalized cross-correlation (NCC)

3. Improve the quality of the aligned image.

Even if you've find a nearly correct alignment, the resulting color image may have a poor quality, i.e. it may suffer from low or high contrast, or the colors may not be so vivid. You can enhance the contrast of the images by applying Laplacian filtering, gamma correction, and histogram equalization, or a combination of those. Additionally, you can choose to work on a different color space, e.g. HSV color space (see function `rgb_to_hsv` of Python `coloursys` module), and accordingly manipulate the appropriate channels to obtain a more brighter image.

Some important remarks:

- Recall that the color filter order from top to bottom is BGR, not the typical RGB.
- Assume that the negatives are evenly divided into 3 plates (i.e., each plate is in exactly 1/3 of the negative).
- Assume that a simple x,y translation model is sufficient for proper alignment.
- In Python, you can easily shift a matrix by using roll function of numpy module.
- For a better matching score, you may ignore the image borders and use only the internal pixels.
- You can devise and utilize an algorithm to automatically crop the borders in order to eliminate the artifacts in the boundaries and obtain a better quality result.

What to Hand In

You are required to submit all your code along with a short report in HTML. For that purpose, prepare a folder containing

- README.txt (*give a text file containing the details about your implementation, how to run your code, the organization of your code, functions etc.*)
- code/ (*directory containing all your code*)
- html/ (*directory containing all your html report, including your images*)
- html/index.html (*html report*)

Archive this folder as **b<studentNumber>.zip** and submit to <https://submit.cs.hacettepe.edu.tr>.

Your HTML report should contain a brief overview of the problem, the details of your approach, and the results of your algorithm on at least 8 negative images with your comments. Show the results of all of the main steps (unaligned result, result after alignment, results of the enhancement). If your algorithm failed to give a satisfactory result on a particular image, provide a brief explanation of the reason(s).

Grading

The assignment will be graded out of 100:

- 0 (*no implementation*), 20 (*an attempt at a solution*), 40 (*a partially correct solution*), 60 (*a correct solution*).
- Good report preparation and your comments on problem and results you obtain will be graded over rest 40 points.

Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode) will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.

References

- [1] <http://en.wikipedia.org/wiki/Prokudin-Gorskii>
- [2] <http://www.loc.gov/exhibits/empire/gorskii.html>