Spring 2018 CS543/ECE549

Assignment 0: Colorizing Prokudin-Gorskii images of the Russian Empire

Due date: Monday, January 29, 11:59:59 PM

This assignment adapted from A. Efros.

Background

Sergei Mikhailovich Prokudin-Gorskii (1863-1944) was a photographer who, between the years 1909-1915, traveled the Russian empire and took thousands of photos of everything he saw. He used an early color technology that involved recording three exposures of every scene onto a glass plate using a red, green, and blue filter. Back then, there was no way to print such photos, and they had to be displayed using a special projector. Prokudin-Gorskii left Russia in 1918. His glass plate negatives survived and were purchased by the Library of Congress in 1948. Today, a digitized version of the Prokudin-Gorskii collection is available online.

Overview

The goal of this assignment is to learn to work with images by taking the digitized Prokudin-Gorskii glass plate images and automatically producing a color image with as few visual artifacts as possible. In order to do this, you will need to extract the three color channel images, place them on top of each other, and align them so that they form a single RGB color image. Some starter MATLAB code is available here, though you are not required to use it.

Data

A zip archive with six input images is available here. **Note that the filter order from top to bottom is BGR, not RGB!**

Detailed instructions

Your program should divide the image into three equal parts (channels) and align two of the channels to the third (you should try different orders of aligning the channels and figure out which one works the best). For each input image, you will need to include in your report the colorized output and the (x,y) displacement vectors that were used to align the channels.

The easiest way to align the parts is to exhaustively search over a window of possible displacements (say [-15,15] pixels independently for the x and y axis), score each one using some image matching metric, and take the displacement with the best score. There is a number of possible metrics that one could use to score how well the images match. The most basic one is the *L2 norm* of the pixel differences of the two channels, also known as the *sum of squared differences* (SSD), which in MATLAB is simply `sum(sum((image1-image2).^2))`. Note that in our case, the images to be matched do not actually have the same brightness values (they are different color channels), so a cleverer metric might work better. One such possibility

is *normalized cross-correlation (NCC)*, which is simply the dot product between the two images normalized to have zero mean and unit norm (see MATLAB function `normxcorr2`).

For Bonus Points

Multiscale alignment. This archive (over 150MB) contains several high-resolution glass plate scans. For these images, exhaustive search over all possible displacements will become prohibitively expensive. To deal with this case, implement a faster search procedure such as an *image pyramid*. An image pyramid represents the image at multiple scales (usually scaled by a factor of 2) and the processing is done sequentially starting from the coarsest scale (smallest image) and going down the pyramid, updating your estimate as you go. It is very easy to implement by adding recursive calls to your original single-scale implementation. Alternatively, if you have other ideas for speeding up alignment of high-resolution images, feel free to implement and test those.

Other improvements. Implement and test any additional ideas you may have for improving the quality of the colorized images. For example, the borders of the photograph will have strange colors since the three channels won't exactly align. See if you can devise an automatic way of cropping the border to get rid of the bad stuff. One possible idea is that the information in the good parts of the image generally agrees across the color channels, whereas at borders it does not.

What to turn in

You should turn in both your **code** and a **report** discussing your solution and results. The report should contain the following:

- A brief description of your implemented solution, focusing especially on the more "non-trivial" or interesting parts of the solution. What implementation choices did you make, and how did they affect the quality of the result and the speed of computation? What are some artifacts and/or limitations of your implementation, and what are possible reasons for them?

- The output color image for every single input glass plate and the displacement vectors that were used to align the channels. When inserting results images into your report, you should resize/compress them appropriately to keep the file size manageable -- but make sure that the correctness and quality of your output can be clearly and easily judged. **You will not receive credit for any results you have obtained, but failed to include directly in the report PDF file.**

- Any bonus improvements you attempted, with output. If you implemented a multiscale solution, report on its improvement in terms of running time (feel free to use an estimate if the single-scale solution takes too long to run). **Any parts of the report you are submitting for extra credit should be clearly marked.**

Turning in the Assignment

Your submission should consist of the following:

- All your code and output images **in a single zip file**. The filename should be **lastname_firstname_a0.zip**.
- A brief report **in a single PDF file** with all your results and discussion. The filename should be **lastname_firstname_a0.pdf**.

The files will be submitted through **Compass 2g**. Upload instructions:

1. Log into **https://compass2g.illinois.edu** and find this course and assignment.
2. Upload **(1) your PDF report** and **(2) the zip file containing your code** as two attachments.
3. Hit **Submit**.

Multiple attempts will be allowed but only your last submission before the deadline will be graded. **We reserve the right to take off points for not following directions.**

Late policy: You lose 25% of the points for every day the assignment is late. If you have a compelling reason for not being able to submit the assignment on time and would like to make a special arrangement, you must send me email **at least a week before the due date** (any genuine emergency situations will be handled on an individual basis).

Academic integrity: Feel free to discuss the assignment with each other in general terms, and to search the Web for general guidance (not for complete solutions). Coding should be done individually. If you make substantial use of some code snippets or information from outside sources, be sure to acknowledge the sources in your report. At the first instance of cheating (copying from other students or unacknowledged sources on the Web), a grade of zero will be given for the assignment. At the second instance, you will automatically receive an F for the entire course.