TNM087 Bildbehandling och bildanalys Reiner Lenz

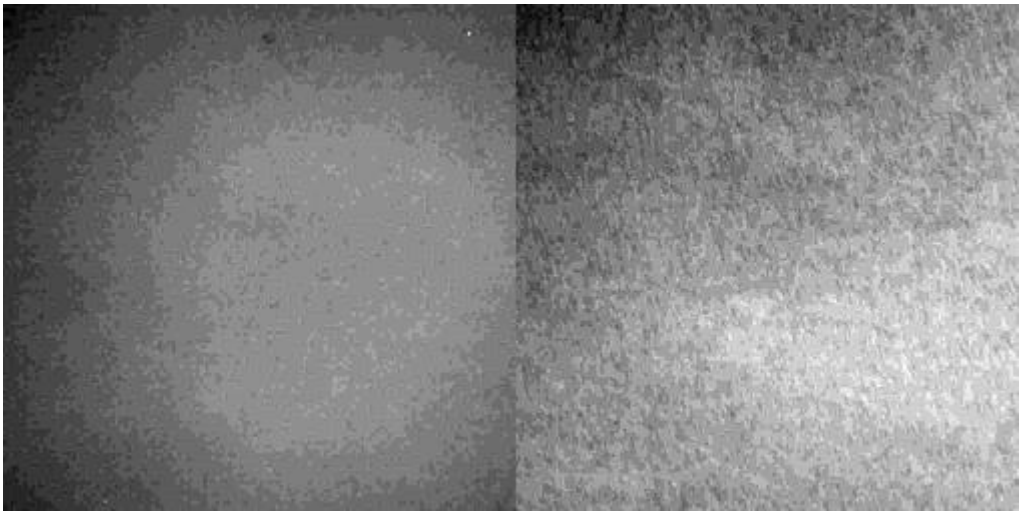
Lab 2: Operations in the image domain

Part 1: Vignette and uneven illumination

Overview:

One often assumes that the illumination is the same at every point in the image and that the camera/optics is perfect. The image of a white sheet of paper should therefore be an image that has the same gray value everywhere. This is never really the case in practice and if one uses the camera as a measurement instrument one has to estimate and compensate these effects before one can use it in the actual application.

The following figure shows the (downsized) versions of the images HWhite1.jpg and CWhite1.jpg that show the same page of white paper under the same illumination conditions and camera settings. The left image was taken with a cheap Holga lens, the right image with a better Canon lens.



You see that the Holga image is darker, and has a visible ring structure: the image is darker in the corners than in the center. This effect is known as vignetting. You can also see that the Canon image is brighter but that the gray values are also varying over the images. This might be an effect of the geometric lamp/camera relation. For more information about vignetting read Section 10.1.3 in the Computer Vision book.

In the first part of this lab you will measure the vignetting effects in the Holga and the Canon image. Hopefully you will find out that the vignetting effect is much more severe for the Holga lens than for the Canon.

Note that one wants to avoid vignetting in cases where the camera is used to measure light. Visually vignetting can be desirable, for example to focus the attention of the viewer to the main object in the

center. This is one of the reasons why Holga cameras and lenses are popular in photography. Search for “Holga” on flickr for more examples (https://www.flickr.com/search/?q=holga).

The goal of this exercise is to measure the average radial intensity which characterizes the vignetting effect.

A. Read in the original images CWhite1.jpg and HWhite1.jpg. Resize them using imresize to a smaller image 512x512. In the following the variable N = 512 denotes this image size. Read the description of meshgrid and cart2pol and explain what you get after the execution of the two commands

   [X,Y] = meshgrid((1:N));

   [T,R] = cart2pol(X-N/2,Y-N/2);

   Plot the vector R(N/2,:) and use imshow(R,[])

B. Scale the elements in the matrix R such that R(N/2-1,1) is mapped to 1. This normalizes the radius variable so that the maximum distance on the axis is equal to 1. The result is a new matrix SR containing the scaled radius values. Now quantize the values in the matrix SR to an integer array with values 0..M where M is (for example) 100. Hint: Use floor/ceil/round to convert the double precision values to integer. These quantized values are collected in the matrix QR. Using Maskm = QR==m produces a logical matrix Maskm with a mask whose values are true where the quantized radius has value m.

   For every value of m compute the following values:

   1. The sum over the pixel values in the image where the mask Maskm is true
   2. The number of object points in the mask Maskm
   3. The average of the pixel values in the image where the mask Maskm is true
   4. The normalized average where you normalize average values such that the maximum value is equal to 1.

      Plot the results of the averages and the normalized averages as curves (at least one for the Canon and one for the Holga lens)

Bonus: (How) Can you generalize this procedure to an angle-dependent description of the vignetting effect?

Bonus: Now that you have a description of the vignetting effect you can use it to compensate, or attenuate, the vignetting effect. You can test it with Holga image from flickr.

Part 2: Correlation

Overview

A common problem in photography is the red-eye effect in flash images. An example is the following image from Wikipedia  http://en.wikipedia.org/wiki/Red-eye_effect .

In the following you will automatically detect the location of the red eyes. The main tool is the Matlab function imfilter which computes correlations and convolutions.

Preparation: read the documentation of the imfilter command in Matlab, filter sections in Gonzalez-Woods and section 3.2 about linear filtering in the Szeliski book.

A) Read in the image BoldRedEye.JPG it should be of size 512x681 pixels. Extract the red channel and call the resulting image RedChannel.
B) Load the file RedEyeMask.mat or generate a circular mask of diameter 32 pixels (use meshgrid). The size of the mask is approximately equal to the size of one eye.
C) Generate a square filter of size 32x32 or 48x48 or 64x64 with constant filter values (either one or 1/NumberOfPixelsInMask) and use imfilter to filter RedChannel with these filters. The result is an image where every pixel value corresponds to the sum (mean) value of the pixels under the filter kernel. Call the result MFilterImage
D) Now filter the RedChannel image with the circular mask from part (B). The resulting pixel value is high when the underlying area is circular and red. This is the image EyeFilterImage.
E) Now compute the pointwise ratio EyeFilterImage/MFilterImage. It has a high value when the matching result in (D) is high relative to the mean value in the window.
F) You can isolate the centers of the eye using the commands imregionalmax together with thresholding using a threshold value computed using quantile.

You can combine the original image and the result image you computed in part (F). Using imshowpair the result should be similar to this image 

Bonus: Try to remove the red eyes in the original image by reducing the contribution of the red channel in the detected regions. Test your code with other red-eye images.

Part 3: Registration

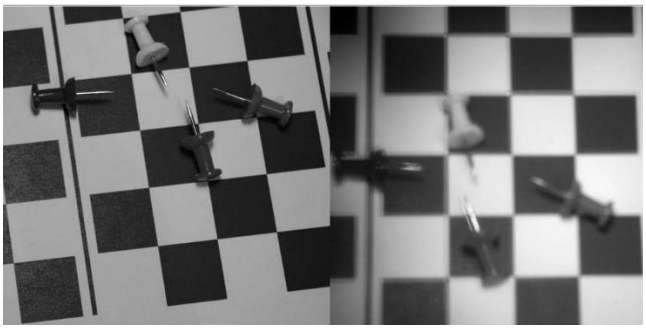
Overview:

Registration is a fundamental operation in image processing: You have two or more images and you want to find common points in several of them so that you can compare them, combine them etc..

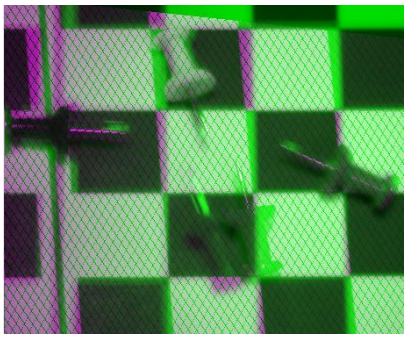
Typical application examples are

- *panorama stitching* where you take several images and combine them later into one big image
- *remote sensing* a satellite is covering the same area at different points in time
- *medical imaging* where you take images of a patient with a CT and an MRI-scanner and you want to combine them for a diagnosis

In this lab you use two different images of pins located on a chessboard-like background



One is taken with a Canon lens and the other with a Holga lens (same as in Part 1). You will compute a geometric transformation that makes it easier to compare them. After the registration the result image should be similar to this image



Preparation: Read the section 2.1 'Geometric primitives and transformations' in the Szeliski book and the description of 'Image Alignment' in Chapter 6 of the same book. Read also the description of ginput, mldivide and mrdive (backslash) in Matlab.

A) Read in the images GCPins512.jpg (called Gim in the following) and GHPins512.jpg (called Him) they show two similar scenes taken with two different lenses: A Canon and a Holga. The goal of the following process is to construct a map of the grid used in Gim, then to modify it and finally map the image Gim so that it becomes as similar to Him as possible: In short find a map A such that Gim(A(x)) = Him(x).
B) Show the Gim image in a figure window and select a number of interesting points with the help of ginput: [GX,GY] = ginput(M);

What is the minimum number M you need?

You may want to mark the selected points for the next step (use text(GX(k),GY(k),numstr(k)) after 'hold on')

C) Next identify the points you selected before in Gim and mark them in the image Him: [HX,HY] = ginput(M)

D) Collect the coordinates of the points from Gim in the coordinate matrix GC of size (M,2), those of Him in HC of the same size. Add a column of ones to both of them so that you have matrices GC1 and HC1 of size(M,3).

E) The unknown coordinate transformation that maps the coordinate vectors in GC1 to the coordinate vectors in HC1 a given by A and we have the matrix equation: GC1*A = HC1

If you have not done it before, read the documentation of the backslash operator *mldivide*

Solve systems of linear equations Ax = B for x; x = A\B;x = mldivide(A,B)

As a result you have a 3x3 matrix A that you can use to compute the three-dimensional coordinate vectors in the Him.

F) Given two images I_1 and I_2 and a coordinate transformation A: $I_1 \rightarrow I_2$ between them you have two options to generate a new image: (1) You can use a point q in I_1 and push it to the point p = Aq in I_2; or (2) You can use a point p in I_2 and pull it from the point $q = A^{-1} p$ in I_1. Use one of them to overlay the two original images.

G) In both cases, push and pull, one of the points has integer coordinates and the other point has non-integer coordinates. Use different interpolation methods (nearest neighbor, linear, bilinear,…) to estimate the pixel value at the non-grid point.