CS 6643 - Computer Vision, Spring 2019 **Guido Gerig**

Assignment 2 Practical

Out: 03 / 05 / 2019
Due: 03 / 28 / 2019 11:55 PM

Late submissions: Late submissions result in 10% deduction for each day. The assignment will no longer be accepted 3 days after the deadline.

Office hours:

		Mon	Tue	Wed	Thu
Guido Gerig	Office 10.094				
Andrew Dempsey	ad4338@nyu.edu	10-12am			
Anshul Sharma	as10950@nyu.edu		10-12am		
Bhavana Ramakrishna	br1525@nyu.edu			10-12am	

Location: cubicle spaces in 2 Metrotech, 10.098 A, B, D, E, H

A) Programming Questions

The purpose of this assignment is to get familiar with implementation
and application of spatial filters for image smoothing, edge detection and template matching.

Hint for implementations: Filtering and edge detection require to use float or double type for image arrays and filter. You can map those images back to integer at the very end of the processing for display purposes, and during this operation you can also optimally scale and shift the result to [0..255] after calculation of minimum and maximum intensities across the resulting image.
Clarification: Color, grayscale and black and white images: Color images have three channels (RGB) and need to be converted into grayscale images before processing. Grayscale images are often also called black and white images (B/W), a term that comes from photography. Binary images, i.e. images with only 0 for background and 1's for foreground, are the extreme form of black and white images with only a range of [0,1].

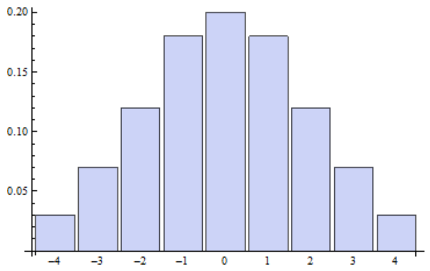
A1) Image Smoothing

Implement a spatial filtering schemes for quadratic, symmetric filters. For simplification, do not filter the boundary as discussed in the course (which means that the boundary of thickness of half the filter size is not filtered).

A1a) Box filtering: Implement a 2D scheme for 2D square filters of size k*k, and design a filter with equal weights (remember to normalize so that the weights sum up to 1.0). Use a k*k smoothing kernel (3*3, 5*5) to reduce noise and smooth input images.
Apply a 3*3 and 5*5 smoothing to our test image b2_a.png. Display images before and after filtering. You may also zoom a subregion to better show the filtering effect (as we have done in the slides).

A1b) Gaussian filtering: Implement a filtering scheme for separable filters (see course slides) where 1D filtering in horizontal followed by vertical direction is applied.

Now let us use a cropped 1D Gaussian filter mask of [0.03, 0.07, 0.12, 0.18, 0.20, 0.18, 0.12, 0.07, 0.03] that corresponds to a Gaussian filter with σ=2.0. Apply this 1D filter first in horizontal direction, save the intermediate results, and then apply it in vertical direction to this intermediate results. This will give you a 2D Gaussian filter with the nice smoothing properties as discussed in the course.



Apply this Gaussian filter as separable filter to the same image as in a) and compare. Discuss eventual differences which you observe between a 5x5 box filtering and the Gaussian filtering.

A1c) Edge detection by Laplacian: The course discussed that edges can be obtained by first derivative filters followed by detection of extrema (non-maximum suppression), or by detection of zero-crossings of the second derivative. We also discussed that edge detection performs best if the original noisy image is smoothed.

0	−1	0
−1	4	−1
0	−1	0

1) Write code for a 2D Laplacian filter with the 2D mask shown here.
2) Use the Gaussian filtered image from b) as the preprocessed smoothed image and apply the Laplacian filtering to this images. Scale the image with the negative and positive regions resulting from Laplacian filtering to a positive range of [0..128] for display purposes, and show the result in your report.
3) Threshold the Laplacian filtered image at 0 so that negative regions appear as "0" and positive as "1". Scale the image to [0..255] for display, and show in the report.
4) (Not required by nice to have: Would you have additional capacity, code a procedure that marks zero-crossings in the binary [0,1] image by detecting and labeling only pixels of "1" (positive region) that have at least one neighbor with "0". This will result in a thin edge map as shown in the course slides.

Hint for possible solution: Sum all pixels in a 3x3 neighborhood, and if result in not 9 there is at least one 0-pixel.

Image b2_a.png from assignment 1

A2) Object detection by Template Matching

Spatial filtering is also used for finding specific objects in images by template matching. Given a template of a sought object, the template acts as a filter, and the maximum correlation indicates the position of the object (see course slides on filtering).

Use the 2D spatial filtering technique implemented before to find image objects. Now, the filter weights are not a user-defined mask but a mask that is represented as a small image template which you select.

A2a): Modify your filtering code so that it can use a small gray scale input image as template, to correlate the source image with this template. Best would be to correlate with a symmetric mask where you determine width (2a+1) and height (2b+1) of the template image, and filter in the range of [-a,a] and [-b,b] with the filter mask centered at [0.0].

A2b): Perform the following steps:

- Threshold the original key image so that the background is [0] and keys appear as [1]. You can use Otsu thresholding or some quick estimate to create this binary image.
- Choose your favorite key. Crop this key with only a narrow white boundary, and save this subimage as your template.
- Modify your template by setting background [0]-pixels to [-1], so that you get [+1] for the key and [-1] for background. Save this new template with signed values as your filter mask. The reason for creating a signed template is significantly improved selectivity.
- Correlate the binary input image with your new template as the filter mask.
- Please recall that this results in a peak (maximum) if the template perfectly matches the specific image region which you selected for your template. The correlation image is signed and covers the range from

negative to positive values. For display purposes, find the most negative and most positive values, scale this range to 255, and shift it to [0..255].

- Do a pass through the correlation image to detect the maximum peak value and its (x,y) location. You may mark this location with overlay of a circle or just manually painting some arrow or similar. Discuss if this location matches your expectation, and also discuss what happens to the peak of the other keys.

In your report, show the original image, peak image, and your template which was used for correlation.

Image and example of selected template:

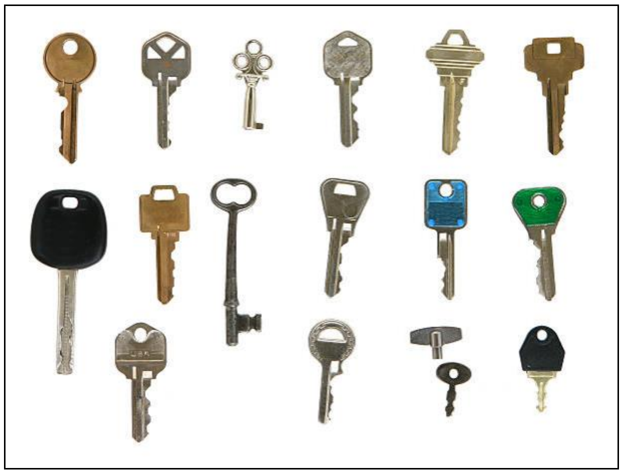
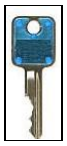


Image "multiple-keys.jpg"



Example Template