Centre for Image Analysis
Swedish University of Agricultural Sciences
Uppsala University

Computer Assisted Image Analysis I

Exercise 1, VT2014: Introduction

The aim of this exercise is to make you familar you with some very basic methods used in computerized image analysis to manipulate and analyze images in the Matlab.

Formalities

- You need an account for the computer systems at the Dept. of Information Technology and for Studentportalen.
- Before you start working on the lab you should sign up for a group in Studentportalen. We recommend that you work together in groups of two people.
- As long as you attend the lab session, this exercise can be corrected orally at the lab session without any written report.
- If you don't finish all the questions or don't attend the lab session, a written report with all the remaining answers has to be handed in as a pdf. The report should be uploaded as a pdf-file to Studentportalen. Please save your report in the following form **Lab1_GroupNumber_LastName1_LastName2**.
- **Deadline:** February 10, 2014.
- The status of your reports will be found in Studentportalen.
- The images and files for the lab can be found in Studentportalen, in the file area Lab material.

1 Getting started

Log on to one of the workstations. Matlab is started by giving the UNIX-command:

```
matlab-7 &
```

Remember to start Matlab from a directory where you have write permission. After a while you will be presented with a graphical interface to Matlab. The files containing all functions and images for this lab can be found on Studentportalen. Functions are ordinary text files, ending with `.m`. They can be viewed either in a text editor or in Matlab's own editor that you reach from the Matlab prompt with `edit <function name>`. You should change the Matlab path to the path that contains functions and images provided by a zip file. The current path is shown if you write `pwd` in Matlab and you can change the path using `cd`.
A useful Matlab command is the `help` command. If you type `help` followed by the name of a standard function in Matlab, valuable information on how to use that function will be displayed. Try for example:

```
help find
```

First you get the syntax, and at the bottom there is a small example of how to use it.

When the paths are set, you can start by reading the image **napoleon.png** by using the Matlab command:

```
I = imread('napoleon.png');
```

The image is now stored as intensity values in the 2D matrix `I`. Since an image is stored as an ordinary matrix in Matlab, all matrix operations available can be used on the image (e.g., addition, subtraction etc).

2 Viewing Images and Saving Images or Figures

There are different ways of displaying image in Matlab. You have the functions `imshow`, `image`,`imagesc` and `imtool` that all display the image in different ways. Function `imshow` shows the image, using the right axis ratio and original graylevel scale of the image. `image` uses Matlabs method for displaying matrices as images. It uses the graylevel scale of the image, but it does not use the right axis ratio. `imagesc` works just as `image` but rescales the graylevels to use the full colormap. To see the color map you can give the Matlab command `colorbar` after you have displayed the image, or you can choose **Insert→Colorbar** in the menu of the figure window. To get a notion how these functions display the image you should try the following Matlab commands:

```
I = imread('napoleon.png');     % read the image
imshow(I);                      % show image I using 'imshow'
figure                          % open a new figure window
image(I)                        % show image I using 'image'
colorbar                        % add a colorbar showing the colormap
figure                          % open a new figure window
imagesc(I)                      % show matrix I using 'imagesc'
colorbar                        % add a colorbar
```

The text written after the `%` is just a comment and will not be interpreted by Matlab.

There is another way of displaying images in Matlab using built in image tool **Image Tool**. Open the image in **Image Tool** by using the Matlab command:

```
imtool(I)
```

where `I` is the image. **Image Tool** will display two windows; the **Overview** window and the **Image** window. The **Image** window has a menu bar and a toolbar with various buttons at the top, a window pane displaying the image in the middle, and a status bar showing various information at the bottom.

To measure pixel values in the image you can move the pointer over the pixel. The status bar shows the current graylevel as:

```
Pixel info: (x, y) f
```

where `f` is the graylevel at the position (`x`, `y`) in the picture. You can also choose **Tools→Pixel Region** in the menu (or by pressing the Inspect pixel values button in the toolbar). This will open the **Pixel Region** window and show a cross hair marker in the image. The **Image Region** window will show the pixels in the part of the image covered by the cross hair marker.

1. ***Where in the image is the pixel*** $(1,1)$ ***located and what is the graylevel value? In command window type*** `I(1,1)`***. Did you get the same value?***

We recommend using `imshow` if you just want to view an image, `imtool` if you want to examine an image thoroughly, and `imagesc` if you want to display a matrix that is not necessarily an image. This will come in handy both during the computer exercises and the project later on in the course.

To save an image you can use the `imwrite` command, but make sure that you are in a directory where you have writing rights.

```
imwrite(I,'my_napoleon.png')        % write image I to file 'my_napoleon.png'
```

use the Matlab `help` if you want more information on how to write images to different file formats. If you want to save an entire figure (including the parts around the image like, e.g., the colorbar) you can use the `print` command.

```
imagesc(I)                          % show image in figure
colorbar                            % add colorbar
print(gcf, '-dpng', 'nap_fig.png')  % write current figure to file 'nap_fig.png'
```

If you prefer using the menu for saving the contents of a figure you can use **File→Save As...**, and choose a suitable file format and filename, to save the contents of the figure. The commands for saving a figure or image will come in handy when putting together your reports for the computer exercises and project later on in the course.

3 Contrast, Brightness and Datatypes

Open the images **napoleon.png**, **napoleon_light.png**, and **napoleon_dark.png** using **imtool** and display their graylevel histograms by selecting **Tools→Adjust Contrast** in the menu (or by pressing the Adjust contrast button in the toolbar). The histogram of the image will appear in the new **Adjust Contrast** window with a pink interval defining the graylevel interval of the image file that is displayed on the screen.

The contrast and brightness of the displayed image can be changed by moving the red handles and changing the length of the pink interval in the histogram. This is only a display function and will not change the values in the image file. Try to change the contrast and brightness of the three Napoleon images so that they all look the same.

2. ***Explain what contrast and brightness are. What change in the histogram is related to the contrast and what change is related to the brightness? Illustrate this by drawing graphs showing how the graylevel transform changes as the contrast and brightness are varied.***

Another way to display a histogram of an image is to use the `imhist` and `hist` commands.

```
imhist(I)                          % show the histogram of I
hist(single(I(:)),256)             % show the histogram of I with 256 bins
hist(single(I(:)),16)              % show the histogram of I with 16 bins
```

Using the `whos` and `class` commans, you can explore the datatypes of your image varables

```
whos                               % explore workspace
class(I)                           % show the datatype of I
```

Common datatypes are `uint8`, `single` and `double`. Some commands for showing images expect a certain range of the pixel values depending on the datatype that is used.

```
Is = single(I)                     % Convert from uint8 to single
imtool(I)                          % This looks OK!
imtool(Is)                         % This looks strange!
imtool(Is/255)                     % This looks OK!
```

Also, be careful when you use `uint8` variables in math expressions.

3. ***Explain the difference of*** `imagesc((I/64)*64)` ***and*** `imagesc((Is/64)*64)`***, where*** `I` ***is*** `uint8` ***and*** `Is` ***is*** `single`***.***

4. ***Demonstrate an expression involving*** `I` ***to make it brighter.***

5. ***Demonstrate an expression involving*** `I` ***to give it lower contrast.***

4 Pixelwise Transforms

Matlab uses the .-prefix to denote element- or pixelwise operators. Use the Napoleon image and try Gamma transformation with different values of g.

```
I = imread('napoleon.png');
imhist(I);
L = double(I).^g;
out = uint8(L .* (255/max(max(L))));
imhist(out);
```

6. ***Use*** $g = 2$ ***and*** $g = \frac{1}{2}$ ***and explain the resulting images.***

Histogram equalization is also a pixelwise graylevel transformation. Continue to work with the three Napoleon images. Perform histogram equalization on them by using the Matlab command:

```
J = histeq(I);
```

7. ***Explain how histogram equalization works in theory. Compare the histograms of the original images and the output images. Do the changes to the histograms and the images agree with the theory of histogram equalization?***

5 Aliasing when Sampling

Open the file **zebra.png**, view it and try to resize it by using the following Matlab commands. Feel free to change the size of the output image by changing 78 to something else when you experiment. Also try to resize the image **cameraman.png** that looks more like a normal image.

```
Jnf = imresize(I, [78 78], 'nearest', 'antialiasing', false);
Jnt = imresize(I, [78 78], 'nearest', 'antialiasing', true);
Jbf = imresize(I, [78 78], 'bilinear', 'antialiasing', false);
Jbt = imresize(I, [78 78], 'bilinear', 'antialiasing', true);
```

The result depends on the interpolation method. The 'antialiasing' option refers to lowpass-filtering of the image that removes high frequencies prior to the down-sampling of the image.

8. ***Explain the role of the interpolation method as well as the role of the lowpass-filter. Which combination of options do you prefer? And why?.***

9. ***Can you give a real-life example of aliasing outside the area of image analysis and signal processing in general? Have you seen this phenomenon before?***

6 Local Filtering

In this task we examine effect of smoothing- and sharpening-filters on the image. To construct a filter you can use following

```
H1 = fspecial(type, parameters);
out = imfilter(I, H1);
```

In **fspecial** function you can choose different types of filters. Test at least three kinds of filters on **wagon.png**, among which there should be at least one sharpening and one smoothing filter. Also examine use of different sizes of the filter masks.

10. ***For each filter, examine the effect of the filter and explain what the filter does to the image.***

11. ***For each filter, how does the size of the filter mask affect the result?***

Median filter is not included in `fspecial` function and to calculate this filter you can use function `medfilt2`. Open the image **wagon_shot_noise.png**. Perform median filtering on the image using different sizes of the filter masks.

12. ***Compare visually the effect of median filtering to the effect of mean filtering and explain the differences on the image wagon_shot_noise.png.***

13. ***What is the advantage and disadvantage of each filter when it comes to the result of the filtering? In general the median filter is more time consuming, why?***

7 Image Arithmetics

Another way of comparing images, that is not only performed visually, like in question 12, is by subtracting two images and examining the difference image. Use the Napoleon image and perform a subtraction of a mean-filtered and a median-filtered version of the image. For a relevant comparison, the images should be filtered using the same original image with the same size of the filter mask.

14. ***Describe the difference image. Is the difference coherent with the answer to question 12? Give cause!***

A SPECT image shows the activity in the brain. When evaluating patient data it is often of interest to make a comparison with an image of a "standard" healthy brain. Standard data is created by averaging over a large number of images of healthy brains. The difference between the standard data and the patient data is found by subtraction.

Images **brain1.png** and **brain2.png** show two SPECT images of healthy brains. Image **brain3.png** shows a SPECT image of a brain from a patient with a stroke.

15. ***How can a "standard" healthy brain, or a mean image, of the two images* brain1.png *and* brain2.png *be constructed?***

16. ***Find the difference between the "standard" brain and the image from the stroke patient (*brain3.png*). Where in the brain is the change located?***

Another form of arithmetic for images is letting a constant value affect the image. Try to add or subtract a constant from an image.

17. ***What happens when a pixel gets a value less than 0 or a value greater than 255? Are there other ways this can be handled?***

8 Geometric Transforms

It is sometimes necessary to geometrically correct images. In object recognition a first step can be to rotate the image so that the object is in a standard position, for example along the vertical axis. Open the image **wrench.png** and rotate it 20 degrees, with and without interpolation, using the Matlab commands:

```
J = imrotate(I,20);
K = imrotate(I,20,'bilinear');
```

where `I` is the wrench image.

18. ***Compare rotations performed with and without interpolation. It is easiest to see differences along lines and edges of the images. What does interpolation mean in this case?***

19. ***In general it is faster to rotate the image by a multiple of 90 degrees than by some arbitrary degree. Explain why. For this task use Matlab functions tic and toc.***

9 Scripting and Looping

It is time to make a small computer program, to learn to automate things and loop over an image. Make a script-file with the extension `.m`, e.g. by the command `edit myscript.m` in Matlab. Run your program on some image that you have downloaded from the internet.

20.
- ***Load the image.***
- ***Convert the image to grayscale using e.g.*** `rgb2gray`***.***
- ***Resize and/or crop your image to*** 128×128 ***pixel size, without changing the aspect ratio of the image.***

- ***Loop (!) over the image using a*** 5×5 ***pixel window. For each such window, compute the average pixel value and store the result in a new image. Treat borders in some controlled way and make sure the result has the same size in pixels as the original image. You will need at least two nested for-loops, to scan rows and columns in your image.***
- ***Subtract the original image from your new filtered image.***
- ***Present the original, filtered and subtracted images in a figure, using e.g.*** `subplot`***.***

And finally, histogram equalization, revisited.

21.
- ***Load the image.***
- ***Convert the image to grayscale using e.g.*** `rgb2gray`***.***
- ***Perform histogram equalization on this image using your own algorithm, without using*** `histeq`***. Efficiency is not so important, you may use for-loops or other ways to compute the transformation and apply it to the image.***
- ***Present the original and equalized images in a figure. Compare with Matlabs native*** `histeq` ***function.***
- ***Make your own histogram equalization into a callable function, e.g.*** `Inew = myhist(I)`***.***

Some hints:

- A histogram can be produced by `h = hist(I, 0:255)`
- `cumsum` is a useful function
- `Inew = T(I)` will replace values in I using T as a lookup table, e.g. `T = [1 4 9 16]` can be used to map 0 to 0, 1 to 1, 2 to 4 and 3 to 9. Try `I = [1 2 3 4]; T(I)`. Observe that all values in I have to integers larger than 0.

That's it. The exercise is done!

22. ***Any comments on the exercise?***