Image Analysis
Exercise 03 - Pixelwise operations

2019

Introduction

In this exercise you will learn to make Matlab programs that perform pixelwise operations.

Data material

The data and material needed for this exercise can be downloaded from

`http://courses.compute.dtu.dk/02502/`.

Start by downloading the data and place it into your Matlab exercise folder.

Exercises

Before starting a new exercise, clear the MATLAB workspace and close figure windows by:

```
close all;
clear;
clc;
```

Initial analysis

First we will be working with an X-ray image of the human vertebra. These images can for example be used for diagnosis of osteoporosis. A symptom is the so-called vertebral compression fracture. However, the diagnosis is very difficult to do based on x-rays alone.

Start by reading the image and inspect the histogram.

```
vb = imread('vertebra.png');
imshow(vb);
figure;
imhist(vb);
```

Is it a bimodal histogram? Can you explain how the different anatomical features are seen in the histogram - or if they can not be seen.

Calculate the minimum and maximum values of the image.

Writing Matlab functions - Histogram stretching

Exercise 1

Take a look at the help for `mean2` and try to understand what it does. Compute the average pixel value of the vertebra X-ray.

Exercise 2

Create a new Matlab M-file called HistStretch.m in your working directory. (use the `edit` function or do it using the Matlab menu).

Create a new function called HistStretch that takes as input an image `I` and returns an image `Io`. It should be called like this:

```
Io = HistStretch(I);
```

The function should return an image where the pixel values in the input image is stretched to be in the interval $v \in [0, 255]$. You should use:

$$g(x,y) = \frac{v_{\max,\mathrm{d}} - v_{\min,\mathrm{d}}}{v_{\max} - v_{\min}}(f(x,y) - v_{\min}) + v_{\min,\mathrm{d}} \ . \tag{1}$$

Explain how to find $v_{\max}$, $v_{\min}$, $v_{\max,\,\mathrm{d}}$, and $v_{\min,\,\mathrm{d}}$.

To ease the computation it is easiest to start by creating a temporary image of type double:

```
Itemp = double(I);
```

This is done since Matlab cannot calculate fractions using integers such as the `uint8` format.

Use your new function on the vertebra image and show the result together with the original image. What changes do you notice in the image? Are the important structures more visible?

Gamma mapping

Exercise 3

Start by plotting the gamma curve

$$f(x) = x^{\gamma} \tag{2}$$

for $x \in [0, 1]$ and for $\gamma = 0.48, 1$, and 1.52.

(hint: use the : operator, and the .^ operator.) Use `hold on` to plot several values of γ in the same figure.

Exercise 4

Create a Matlab M-function `GammaMap` that takes as input an image and a value for γ and returns a gamma mapped image.

Remember to scale the pixel values to $v \in [0, 1]$ before the mapping and back to $v \in [0, 255]$ after the mapping. The mapping should be done on an image of type `double`.

You can scale the image both by just dividing by 255 or by doing a linear scale with $v_{\max,\,\mathrm{d}} = 1$, and $v_{\min,\,\mathrm{d}} = 0$. Try to explain what the difference is.

To be able to show the transformed image it is easier to convert the image back into type `UINT8`.

Use your new function on the vertebra image with different values of γ and show the results together with the original image. What do you see? Did it improve or degrade the image?

Matlab function `imadjust`

Exercise 5

Read the documentation for the Matlab function `imadjust`. Try to use it on the vertebra image with different parameter values. Can you find some values that works well with the given image?

Thresholding

Exercise 6

Create a new Matlab M-function called ImageTreshold that takes as input an image and a threshold value T and returns an image where background pixels have value 0 and foreground pixels value 1.

Test the function on the vertebra image. Is it possible to find a threshold that separates the vertebra from the background?

Exercise 7

Read the documentation for the Matlab function `graythresh` and try it on the vertebra image.

Use the found threshold together with your own threshold program. What threshold value was found? Compare the value with the histogram and try to explain the value.

Exercise 8

Use your camera to take some pictures of yourself or a friend. Try to take a picture on a dark background. Convert the image to grayscale and try to find a threshold that creates a *silhouette* image (an image where the head is all white and the background black). Alternatively, you can use the supplied photo `dark_background.png`.

Matlab Image Tool

Matlab has a simple interactive image tool. It can be used to do simple manipulation of images.

Exercise 9

Open an image with the image tool. For example the vertebra image or your own image.

```
vb = imread('vertebra.png');
imtool(vb);
```

use the `adjust contrast` menu option and play around with the limits.

Color Thresholding in RGB space

In the following we will make a simple system for road-sign detection. Start by reading the image `DTUSigns2.jpg`. We want to make a system that do a *segmentation* of the image - meaning that a new binary image is created, where the ones in the image correspond to the sign we want to detect.

We do that by tresholding the colour-channels individually. This code segments out the blue sign:

```
im = imread('DTUSigns2.jpg');
imshow(im);

Rcomp = im(:,:,1);
Gcomp = im(:,:,2);
```

```
Bcomp = im(:,:,3);
% Create a binary segmentation image based on the RGB components in the original image
segm = Rcomp < 10 & Gcomp > 85 & Gcomp < 105 & Bcomp > 180 & Bcomp < 200;

figure;
imshow(segm);
```

Now make a program that can find/segment the red DTU sign. You can use the `Tools | Data Cursor` from the Matlab figure menu, to inspect the RGB values in the relevant parts of the image.

Color Tresholding in HSI/HSV space

Sometimes it gives better segmentation results when the tresholding is done in HSI (also known as HSV - hue, saturation, value) space. Start by reading the `DTUSigns2.jpg` image and convert it to HSV:

```
HSV = rgb2hsv(im);
imshow(HSV)
Hcomp = HSV(:,:,1);
Scomp = HSV(:,:,2);
Vcomp = HSV(:,:,3);
```

Try to visualise each component (H, S, V) in separate gray scale images.

Now make a sign segmentation tool using tresholding in HSV space and locate both the blue and the red sign.