LUNDS TEKNISKA HÖGSKOLA
MATEMATIK

TENTAMENSSKRIVNING
Image Analysis
2017 October HOME EXAM

Home Exam

You may use any books and computer programs (e.g. Matlab, Maple, Python, Julia, etc.), but it is not permitted to get help from other persons. It is ok to use existing implementations of known methods, but the methods used should be explained in detail. Note that since the exam is only for grade 3 to 5, the normal scale (3 points = grade 3, 4 points = grade 4 etc.) will **not** *necessarily be used. You can pick up the exam from Friday 20 October to Friday 27 October from the course secretaries on the fifth floor of the math building. The exam should be sent, at the latest* **48 hours after pick up** *electronically* **as a pdf-file** *by e-mail to* `kalle@maths.lth.se`*. You may also send in your program-files, but the grading will primarily be done from reading the pdf-file. Contact* `kalle@maths.lth.se` *for the oral exam.*

1. In the two variables `pointx` and `pointy` the respective $x-$ and $y-$coordinates for a number of points on a number of lines are stored. They are corrupted by both outliers and random Gaussian noise. Use RANSAC to fit a number of lines to the data. How many lines are there?

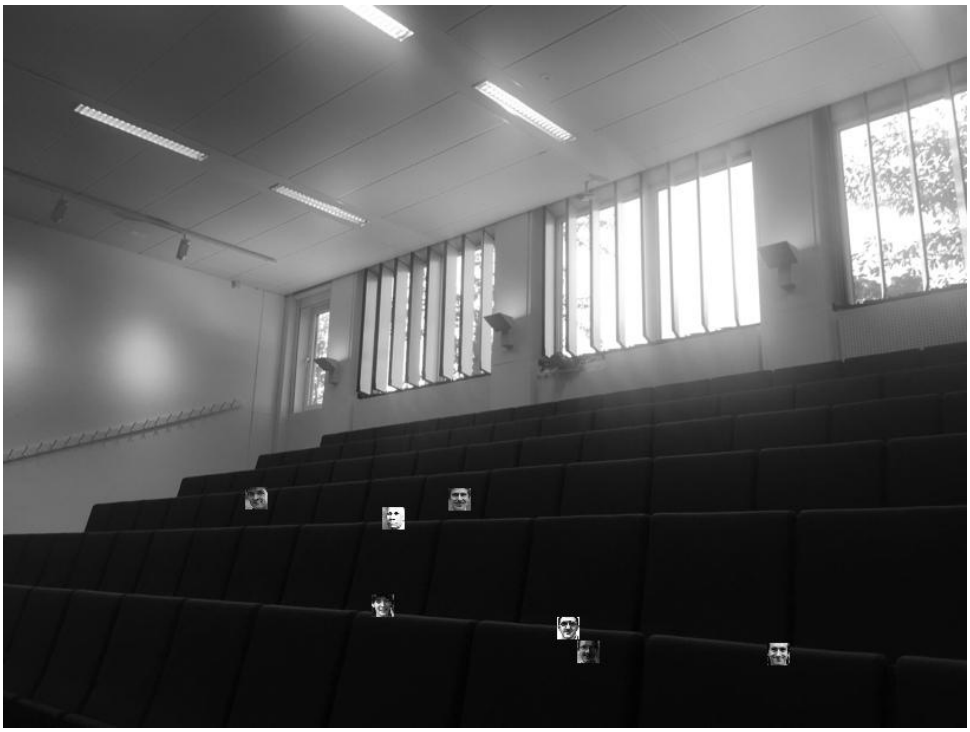
2. In the variable `moon` a dark image is stored. It contains real measurement of the moon counting the number of photons hitting the detector. Try to enhance the image in terms of contrast and noise. Preferably so that one can see (i) the background, (ii) the part of the moon that is lit directly by the sun and (iii) the dark side of the moon. The dark side of the moon is actually lit indirectly from the sun, by light that are reflected from earth.

3. In a Sudoku the object is to fill in a 9×9 square with the numbers $1 \ldots 9$ so that in each row and each column and each 3×3 subsquare, there is exactly one of each number. Instead of using numbers one can use any kind of symbols. In the variable `symbolsimage` nine symbols are shown. In the variable `sudoimage` a corresponding Sudoku is shown. Construct a method that converts this image to an ordinary Sudoku and solves is. For solving the actual Sudoku you can e.g. use
`http://www.mathworks.se/matlabcentral/fileexchange/41884-sudoku`

4. An image of blood cells is given in the variable `blood`. Use methods from mathematical morphology (or some other method) to construct an algorithm that counts the number of cells in such an image.

5. In the variable `inpaint` an image is stored. Some text has by mistake been placed over the image. Try to remove the text from the image automatically.

Figur 1: The image 'inpaint'. Remove the text.



Figur 2: The image 'ma06'. Find the faces.

6.

Prelude

In the variable `ma06` a grayscale image is stored. Study the image. The image is of the lecture hall MA:06, where a number of faces have been added synthecially.

Part A

In the variable FaceNonFace that you used in assignment 2 earlier, there are 200 examples of 19×19 images of either faces or not faces. In the variable X these are column stacked so that each column is a 169×1 vector x containing the column stacked version of a 19×19 image. Use any package to train a linear logistic classifier. so that

$$y = w^T x + b,$$

is a value that after applying the logistic function

$$z = s(y),$$

can be interpreted as the probability of the image representing a face. Here the logistic function is

$$s(y) = \frac{1}{1 + e^{-y}}$$

as usual.

From the image `ma06`, extract a 19×19 cutout image X centered at row 435 and column 331. Plot this cutout-image X. Make a column vector representation x of this 19×19 cutout X. Calculate the values y and z for this cutout-image-vector x. Do the same for a cutout centered at row 123 and column 456.

Part B

How can the convolution be used to calculate the value y for each 19×19 cutout centered at each and every pixel position? Don't worry that there will be errors close to the border of the image? (Hint: It is possible to implement this in matlab with a few commands such as `reshape` and `conv2(?,?,'same')`. You might also have to use `flipud` and `fliplr`)

Calculate 'z' pixelwise from 'y' above. Plot them, and check that you get the same result at positions $(435, 331)$ and $(123, 456)$, as compared to your calculations in part A.

Part C

Use, for example thresholding and non-maximum suppresion of $|z|$, to find the center of the faces. Plot the detected faces as a small square around the detected center points.