הבינתחומי הרצליה
IDC HERZLIYA

Computer Vision
Assignment 1

Fall 2018, Yael Moses

Submission date: 11.11.2018

- Read carefully the instructions
- Submit using moodle:
- Submit on time!
- You can submit this assignment in pairs.
- Please go over warm_up_Matlab – it will help you!
- Make sure the code run
- Look carefully on the instructions for submission.

Section A: Convolution (15pt)

a. Suggest a convolution mask of size 5×5 such that the maximal value over any image will be obtained in a region that consists of:
   1×3 white patch on background.

b. Suggest a convolution mask that computes the average of a 1×9 pixels around each pixel.

c. Suggest a convolution mask of size 5×5 such that the maximal value over any image will be obtained in a region that consists of:
   A white '+' shape (vertical and horizontal lines of length 3 of width 1) on a black background.

The scrips under Sectpion A should contains:
For each maks:
- The mask you suggest
- A code that generate a synthetic image to demonstrate the results.
- Display the results of the convolution of the image with the mask
- Mark on the original image the max value of the convolution.

Useful Matlab functions:
- conv2: (use 'same')
- *a=zeros(100,100); % generates a black image*
  a(1:10,:)=255; %genreates a white rectangle on a blac
- *figure; imshow(a)*
  hold on;
  *plot(x,y,'*r') % plots a red * on the figure on the last figure*
- *Use imshow(B,[])* % Matix B contains values that maybe negative or % non necessarily integers between 0 to 255.

A question for thought (not for submission):

- What is the affect if you do not normalize the mask such thaht the sum of elements is one?

Section B: Canny Edge Detector (40 pt)

In this assignment, you will implement the classic Canny edge detector, evaluate the results, and answer some questions. You are not allowed to use any available existing Canny edge code.

Reference:

- F. J. Canny. A computational approach to edge detection. IEEE Trans. Pattern Analysis and Machine Intelligent (PAMI), 8(6):679-698, 1986.

A. Write the function: *E=canny('file_name', sigma ,L_th, H_th)*

Use the following steps:

1. Compute two masks with the derivative of a Gaussian. The parameters of the function are sigma and size.
   a. *G_dx=Deriv_Gauss_x(sigma,mask_size)*
   b. *G_dy=Deriv_Gauss_y(sigma,mask_size)*

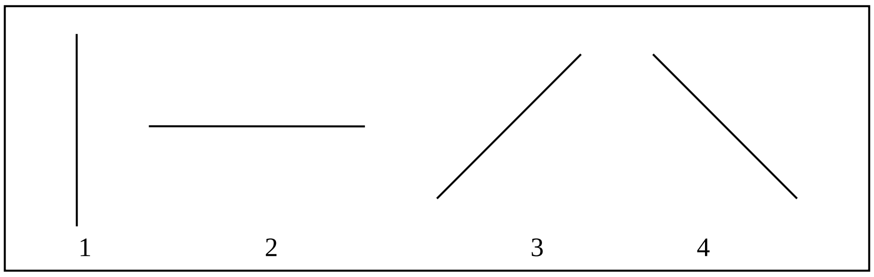
   Note:
   - You can look at the warmup matlab for a similar function that generates a Gaussian mask.
   - The mask_size should be about twice the value of *sigma*. You should check for yourself that the mask_size is appopriate (not too large and not too small) by looking at a section through the middle of the mask.
   - **A question for thought (not for submission):** Do not answer: should you normaize the sum of the elements to be one in the mask as in Section A?

2. Using these masks compute two matrixes, *I_x* and *I_y*, with the derivatives of the image in the *x* and in the *y* directions, respectively. Compute two matrices *G_orientation* and *G_magnitute* with the gradient orientaion and magnitute at each pixel.

   Note:
   - You can do it with matrix element-by-element operations (using '.*' , '.^', etc.)

3. Compute non-maximum suppression into a matrix *Et=thinning*(*G_strength,G_orientation)*.

Note: For computing non-maximum suppression, edge orientation should be rounded to be one of four orientations:
Gradients that are approximately horizontal, approximately vertical, and approximately one of the diagonals (see figure).



4. Then use the two thresholds, *L_th, H_th,* to put it all together and compute the canny edge detector.

Submit: the matlab functions.

B. Create a synthetic test image, im_test, that can be used to verify that all parts of your edge detector, works properly. Explain why you choose this image. Run your edge detector on this image.
- You can use meshgrid to generate a function of your choice.

Submit as part of the script: the function that create the image, and a call to this function.
Submit in the doc file: A short justification for your choice.

C. Run it on any image you choose, play with the parameters so it will look “good”.
Submit: the grey level image you choose.
Submit as part of the script: the call to the canny and the call to display the result.

D. Run your Canny edge detector on the three given images:
Church.jpg, Golf.jpg, Nuns.jpg.

E. Explore with different sets of parameters *sigma ,L_th, H_th.*
Answer: Explain how each of them affect the results.

Submit as part of the script: a script that run the edge detector with these sets. Example:

```
%% C.
%% The sigma parameter affects the  …
%% This can be demonstrates by running the canny edge detector
with …
```

```
%% the following two (or three) values of  sigma …
E1=canny('file_name',sigma1,th_low,th_high)
E2= canny('file_name',sigma2,th_low,th_high)
Figure; imshow(E1,[]);
title('Using sigma1')

Figure; imshow(E2,[]);
title('Using sigma2')
```

Section C: Evaluation (30pt)

F. Evaluation measure
To measure the performance of an edge-detector, it is common to compare the detection results with those chosen manually by a person. The comparison is between two sets, the set E of pixels detected as edges and the ground truth (GT) set of pixels selected manually. To compare the sets, it is common to use two measures, Precision (P), Recall (R), and F measure, which combines the two measures:
$P = \frac{|E \cap GT|}{|E|}$, $R = \frac{|E \cap GT|}{|GT|}$, $F = 2\frac{PR}{P+R}$.

Write the function**:** [P,R,F]=evaluate_naive(E,E_GT)
Submit: the matlab functions

a. **Answer:** Which choice of Canny's parameters will cause P to be high?
b. **Answer:** Which choice of Canny's parameters will cause R to be high?
c. **Answer:** Why do we need the F measure as well?

G. Use the given GT results (Church_GT.bmp, Golf_GT.bmp, Nuns_GT.bmp) on the three images, and compute the P, R, and F values for a set of 27 parameters (a combination of 3 values for each parameters**).**
Note:
- These are binary images, so to view them use
  im=imread('Church_GT.bmp')
  figure; imshow(im,[]);

H. Submit as part of the script:
The list of the 3 values you considered for each of the parameters, and display the best set for each of the measure for each of the

images.
Observe:
- Do you have the same set of parameters for the three measures (P,R,F) for the same image?
- Do you have the same set of parameters for all images for a given measure?

I. The computed edges can be shifted by one or two pixels due to digitization. Write a matlab function [P,R,F]=evaluate(E,E_GT) that is tolerant to such a shift.
Note:
- You should use *imdilate* matlab function.

Submit: the matlab functions
Answer: Are the result (P,R,F) remain the same? Are the best set of parameters remain the same with the new function? Give a short explanation.
You can consider here only one of the images.

Section D: Comparison (15pt)

J. Implement the SOBEL edge detector, run it on the three images.
K. Explained the places where SOBEL fails while Canny edge detector succeeds.
Submit: the matlab functions
Submit in the doc file: The explanation.

L. Repeat **(G)** with a Sobel edge detector – there is only one parameter there. You can consider here only one of the images.

M. **Answer:** Which edge detector gives the highest F measure? Give a short discussion of where in the image they give different results.

Hand in:

A zip file containing the following files:

A. All answers to all questions should be written in the report.
B. **Submit a Matlab script file** answer_sc.m that will run all your code and answers. **Make sure the code can run!**
An example file that can help you start is attached. Please use it.
Use titles for each figure so the TA can run what you were asked to put in the script.
The script should have comments for what you present there.
For example, in section B QB:
% Following is the image generated for testing the edge detector:

C. Make sure to use title('') on each figure you generate, so we can understand what you present in that figure.
D. A word document, named HW1-YourName.doc. The word document should include:

    a. Your names.
    b. Answers to all questions.
    c. Few words on how to use your functions.

E. The code with **documentation**. Must include:

    a. answer_scr.m
    b. All the files with the matlab functions you wrote.

F. The additional image we asked you to provide.

More useful Matlab functions:

- Matlab Tutorial: there are plenty on the web.
- Have a look at *meshgrid.*
- Have a look at *conv* and *conv2* and consider using the parameter *'same'* of these functions.
- Have a look at the function *find.*
  A useful example: *a(find(a<50))=0*
  This command sets to 0 all values of a matrix *a* that are smaller than 50.
- Look at the function *sum*.
- Look at the function *bwselect.*
- Avoid as much as possible the use of loops.

Please do not hesitate to contact me if you have any problems.

Good Luck!