Computer Vision - Second Graded Exercise Session 22/05/2014

Instructions

You have **1h 45 minutes**. **You have to submit your files by 10.00AM.**
Any late submission will result in a penalty of -5 points (total is 100 points) for every minute.

Notify one of the assistants before leaving the room.

Once you are done, make sure that your files are successfully uploaded to the system by checking with one of the assistants.

Access to Internet resources is forbidden.

There are two exercises, and they do not depend on each other. If you get stuck in one of them, you can proceed to the other. In all, you are asked to implement the following three functions:

1. `smoothing.m`, for Task 1A.
2. `superpixel.m`, for Task 1B.
2. `thresholding.m` for Task 2A.
3. `adaptive_thresholding.m` for Task 2B.

The files for each task have already been put in separate folders, namely Task1 and Task2.
In each folder there is a file called `main.m` that sets up necessary data and runs the experiments.

You do not have to modify any other code but the one in the files mentioned above. Also note that, we will look into your code and give grade, so if you implement something that doesn't yield the final results, you will still get some grades for the part you have implemented, if they are correct.

Task 1 - Superpixels

→ Task 1A

You are given the image shown above. You are asked to implement a function to smooth the RGB image (keeping all 3 channels). Use gaussian smoothing with the given parameters. This function need to be implemented in the `smoothing.m file.`

***Hint**: you can matlab's built-in functions 'fspecial' and 'imfilter'*

→ Task 1B

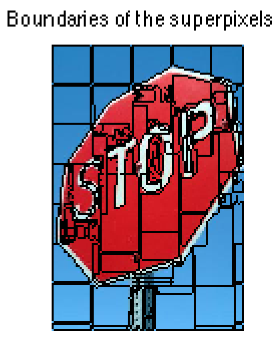
In the `superpixel.m` you are now asked to implement the superpixel algorithm, which works on grayscale image (i.e., 2D matrix). This algorithm is similar to clustering. What you need to do is to combine pixels in the image that are close to each other in terms of distance and intensities. Note that, in the `main.m` we have converted the RGB image to grayscale, on which you run the superpixel algorithm. To do so you are suggested to implement the following steps:

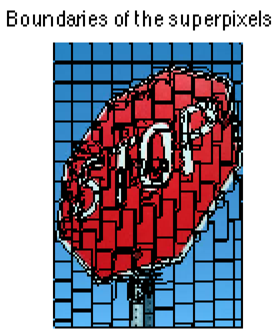
1. Create a `mask` of the same size as the 2D grayscale image. Each pixel of the mask contains an ID of the superpixel it belongs to, or zero if it is not assigned to any superpixel yet. Initially, fill all pixels of the mask with zero.
2. Start from the pixel i, j = (1,1) of the image; set `superPixelCount` = 1;
3. Find all the pixels that are 'close' to it, namely the pixels that fulfill simultaneously the following three conditions:
    a) abs(im(y,x) - im(i,j)) < diff (intensity comparison)
    b) (x - j)^2 + (y - i)^2 < sz^2 (distance in the image)
    c) mask(y,x) == 0 (not assigned to any other superpixel)

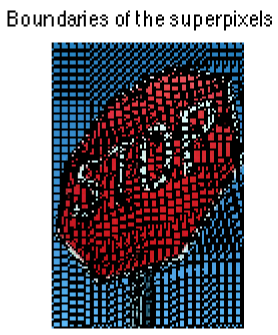
4. For each of those pixels, set their value in the mask to `superPixelCount` to flag that they belong to a new superpixel.
5. Increment `superPixelCount` by 1.
6. Go to the next pixel (i,j), for which the value of the mask(i,j) is equal to zero;
7. Repeat the steps 3 - 6 until all the pixels in the mask have value 1;
8. Return the mask image, which should have all its values different from zero by now.

Hint: *you can use `find()` and `meshgrid()` if necessary.*

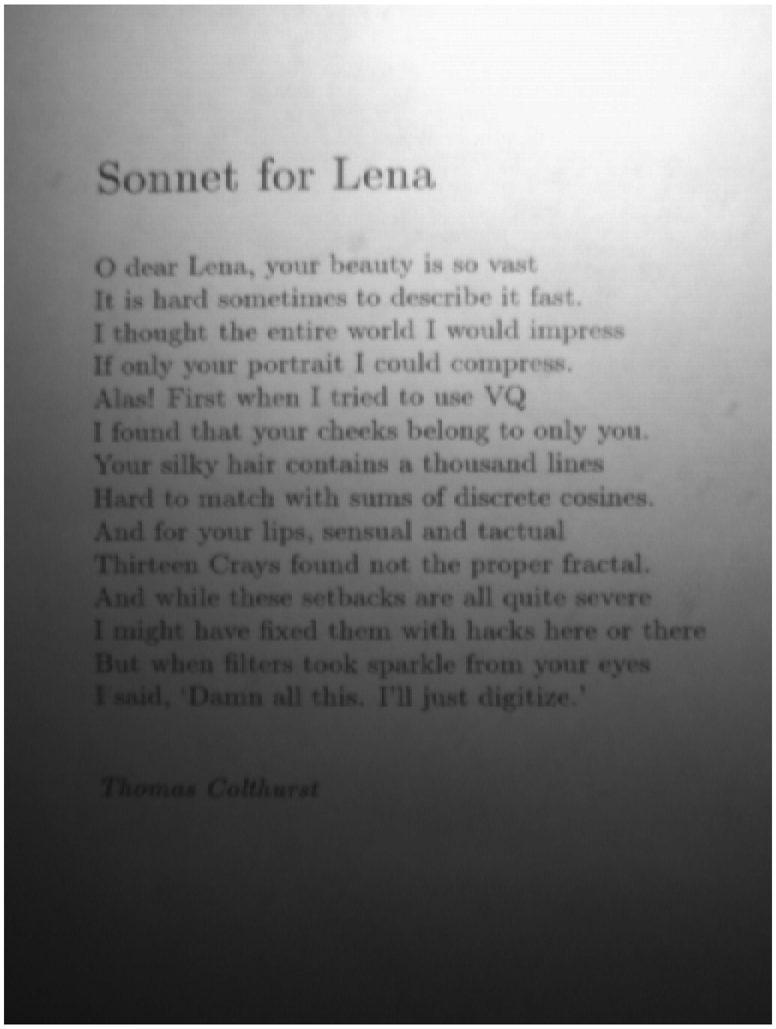
Once you have implemented the steps above, you will see the following output, with superpixels of different sizes. The black boundaries represent the boundaries between adjacent superpixels. It is fine if you get rounder boundaries.







Task 2 - Adaptive thresholding



Sonnet for Lena

O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and tactual
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this. I'll just digitize.'

Thomas Colthurst

→ Task 2A: Basic Thresholding

In `thresholding.m` you are asked to implement the simple thresholding algorithm to separate white background of the sheet of paper. The simple algorithm involves estimating the best threshold that separates the gray levels in the image, and making a mask, so that those pixels, whose gray level is higher than a threshold are assigned 1, and the rest 0. Note that, in `main.m`, we have converted the RGB image to grayscale, on which you will run the thresholding.

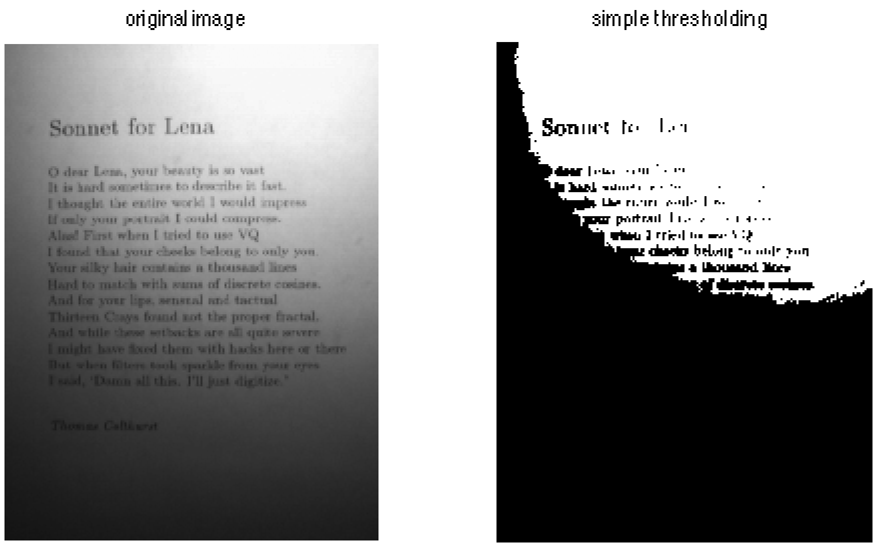
You have to implement Otsu's algorithm for estimating the optimal threshold:

1. Make a histogram of the intensities (**h**) of the image. The histogram has (**N**) number of bins, passed as a parameter.

***Hint**: you can use matlab's built-in function 'imhist'*

2. For every bin **i** of the histogram, calculate its probability **p(i)**, which is equal to the value of the bin in the histogram divided by the sum of the values of all bins in the histogram. ***Hint**: to check that you implemented this correctly, the sum of all p(i) should be 1.*
3. For every **t** from 1 to **N** calculate compute ω_1 as follows: $\omega_1(t) = \Sigma_{i=1}^{t} p(i)$
4. Then for every **t** from 1 to **N** calculate the mean: $\mu_1(t) = \left(\Sigma_{i=1}^{t} p(i) i\right) / \omega_1(t)$
5. Use the following formulas to estimate ω_2 and μ_2 for every **t** from 1 to **N**:
$$\omega_2(t) = 1 - \omega_1(t);$$
$$\mu_2(t) = \left(\Sigma_{i=1}^{N} p(i) i - \Sigma_{i=1}^{t} p(i) i\right) / \omega_2$$
6. Find the value of **t** that maximizes the following function:
$$t = argmax(f(t)),$$
where $f(t) = \omega_1(t)\omega_2(t)\left(\mu_1(t) - \mu_2(t)\right)^2$
7. To find the optimal threshold by normalizing **t** in the following way:
$$thresh = \frac{t-1}{N-1}$$
8. In the same function you now need to create a mask, of the same size as the input image, which has 1 for the pixels for which the intensities of the original image are higher than the threshold, and 0 for the others.

Once you have implemented these steps you should see the following:



Sonnet for Lena

O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and tactual
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this. I'll just digitize.'

Thomas Colthurst

(you may also get a result that is the inverted version of the image on the right, that is correct as well)

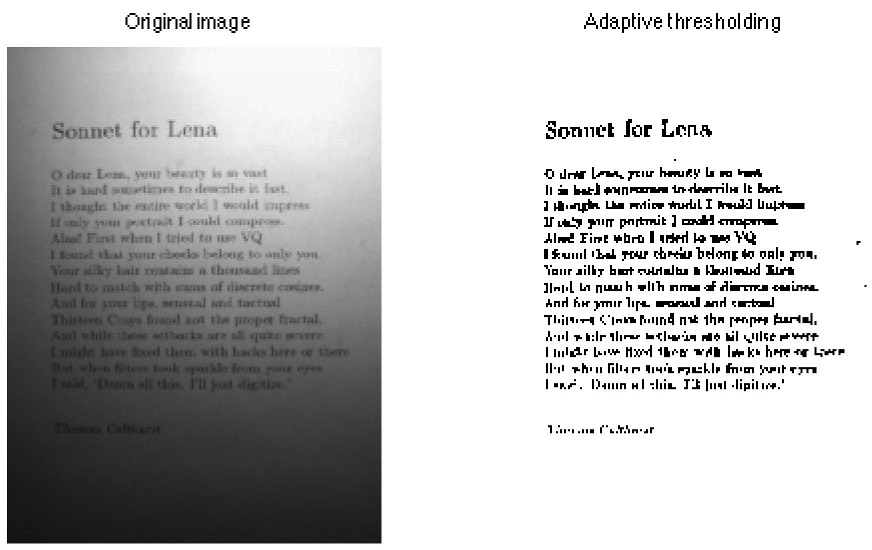
--> Why is performance so poor? **write your explanation as comments in** `thresholding.m`

→ Task 2B: Adaptive Thresholding

In this task, you are asked to implement simple adaptive thresholding algorithm in `adaptive_thresholding.m`. Similarly, you will work on grayscale image. You are suggested to implement the following steps:

1. Convolve image with an averaging filter of size (**sz**), passed as an argument to the function. Don't forget to correctly process the boundaries.
   Hint*: you can matlab's built-in functions 'fspecial' and 'imfilter'*
2. Subtract the original image ***from*** the convolved image that you acquired at the previous step.
3. Subtract the threshold from the resulting image. This threshold is passed as the third parameter to the function `adaptive_thresholding.m`.
4. Create a mask of the same size as an input image, which has the values equal to 1 for those pixels, where the resulting image from the previous step is more than 0 and 0 otherwise.

Once you have implemented these steps you should see the following:



Sonnet for Lena

O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and tactual
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this. I'll just digitize.'

Thomas Colthurst

(you may also get a result that is the inverted version of the image on the right, that is correct as well)