Centre for Image Analysis
Swedish University of Agricultural Sciences
Uppsala University

Computer assisted image analysis I

Excercise 3: Segmentation and analysis of circular objects using
MATLAB Image Processing Toolbox

November 2020

The aim of this exercise is to let you experiment with a number of functions from the MATLAB Image Processing Toolbox. Your task is to create a short script that will count and measure the size of a number of objects in an image. To complete the task, a number of functions have to be combined. There are several ways to solve the problem and there is no "perfect solution".

Formality

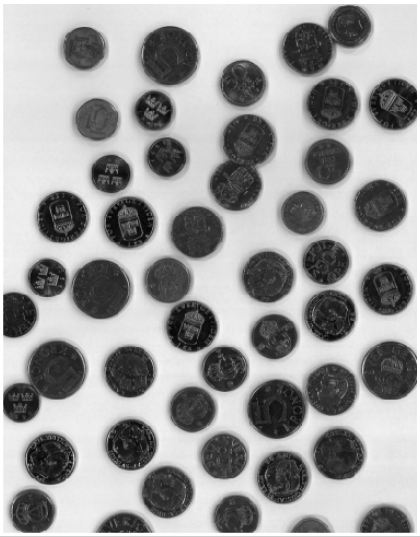
- We **strongly** recommend that you work together in groups of two or three people. This way you can get help quicker, and have someone to discuss the lab with.
- The exercise is corrected by handing in a written report on Studium. Entitle your report **Lab2_LastName1_LastName2**.
- **Deadline:** December 11, 2020.
- The status of your reports will be found in Studium.

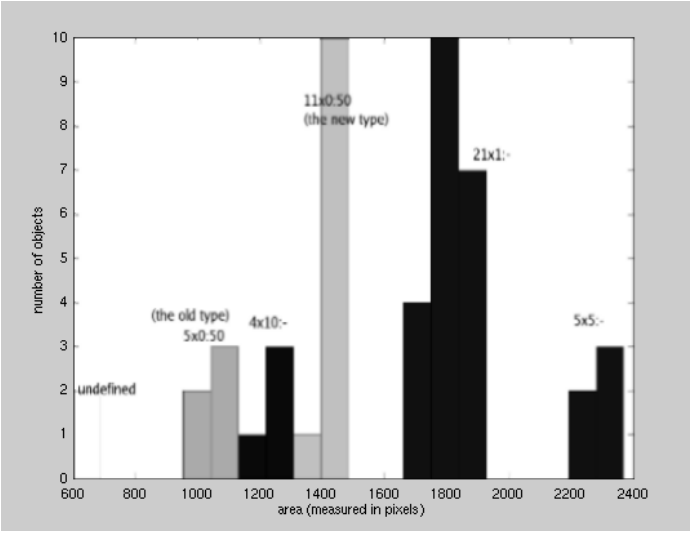
1 Getting started

Copy the image `coins.tif` to a directory where you have write permission, and start MATLAB from the same directory by typing `matlab`. Open the image by typing `I=imread('coins.tif');` and look at the image by typing `imshow(I);`

2 Assignment

Counting objects is a fairly common problem in image analysis. Often the segmentation is difficult, due to non-uniform illumination, shadows, blur, reflections etc. The problems here are that some objects are slightly overlapping or partially hidden (but still possible to identify). Some of the objects also contain bright pixels





that should be counted as part of the object. Those difficulties must be taken into consideration when solving the assignment.

The report should contain:

1. Your script that solves the task. Include a comment for every function saying why that particular function was needed.
2. An image showing the final segmentation result.
3. A histogram showing the distribution of object size. Either area or radius could be used as a size measure (see figure).
4. A discussion of errors and limitations in your final method.

5. As you can see, the objects are coins. Is it possible to count the total amount of money using your algorithm?

6. Explain how your method treat the coins on the image border.

7. Is your solution general in the sense that it can be used when analyzing images with arbitrary circular objects (i.e. not only coins.tif)? Try it on `bacteria.tif`. Does it work on arbitrary ellipsoid or convex objects? Describe some possible modifications you could make to your program.

3 Some useful functions

To get familiar with the tools you need to complete the task, you can try the functions listed below. Use the MATLAB Help window to find the details of the functions and their use. All listed functions are not needed to solve the task, but you should be able to solve the task with these functions only. You are however very welcome to use any other functions you may want to try.

It could be a good idea to start working with a small sub-image of **coins.tif**, for example **Ismall=I(1:50,301:350);** This will make it easier to look at the actual intensity values to figure out what the functions do. The intensity values are shown if you just write the image name, leaving out the **;**.

- **`im2bw(I,T);`** Threshold the image at level **T** ($0 \leq$ **T** ≤ 1) , creating a binary image.
- **`graythresh(I);`** Find a suitable intensity threshold based on the image histogram.
- **`Idist=bwdist(I,'type');`** Distance transform a binary image using different types of metrics. To view the result, use `imshow(mat2gray(Idist));`.
- **`watershed(I);`** Watershed segmentation. Can be applied to the distance transform.
- **`and(A,B), or(A,B), xor(A,B), not(A);`** Logical operations between binary images.
- **`Ilabel=bwlabel(I,nb);`** Labeling of a binary image. nb is the neighborhood, and can be 4 or 8 in 2D. The result can be viewed as a color image writing `imshow(label2rgb(Ilabel, 'spring'));`
- **`F=regionprops(Ilabel, 'feature');`** Extract different features from an image with labeled objects. See the MATLAB Help for description of features. If the feature is 'Area', an array of area values is created by writing `A=[F.Area]`, and a histogram is created by writing `hist(A)`.

If the distance transform is used for segmentation, its maxima can be extracted as a measure of maximum radius. This measure will give a correct measure of radius also for partly overlapping objects and objects cut by the image borders.

- **`imextendedmax(I,h);`** Finds local maxima of height h as compared to the local surroundings. If h=1, the local maxima will be found. The Extended-maxima transform can, for example, be applied to the distance image.
- **`If=imfilter(I,f);`** Filters a 2D image (I) with a filter f. To smooth the image, a 3x3 mean filter can be created writing. `f=[1 1 1;1 1 1;1 1 1]./9`; There are 4 ways of dealing with filtering at the boundary, 'X' (values outside the bounds of the image are assumed to have the value X), 'symmetric','replicate' and 'circular'. When no boundary option is specified, imfilter uses X = 0.
- **`medfilt2(I);`** Median filtering using a 3x3 neighborhood filter.
- **`imopen(I,se), imclose(I,se), imdilate(I,se), imerode(I,se)`** Open, close, dilate or erode the image using a structuring element **se**.
- **`se=strel('shape', parameters)`** Create morphological structuring element **se**.
- **`bwmorph(I,'operation');`** Implementation of a number of different morphological operations, such as skeletonization, top-hat transformation etc.

You can find more information about morphological operators in **Morphology_examples.pdf**.

Good luck!