Image Thresholding

In the previous exercises, we detected edges and we investigated whether these edges create a line. Today, we will detect whole areas in images. The presented method, called image thresholding, expects that objects in images have a similar brightness or color. If this condition is true, we can threshold the image such that we get a binary image - the object pixels have the value 1, the other (background) areas have the pixel value 0. These binary values are assigned according to the chosen threshold T. If a pixel value is higher than T, the image value is set to one, zero otherwise (or vice versa, depending on the segmentation task). Written as equation, it is

$$g(x,y) = \begin{cases} 1, & f(x,y) > T \\ 0, & f(x,y) \leq T \end{cases} \tag{1}$$

where $g(x,y)$ is the output image, $f(x,y)$ is the input image. The value of T is the only parameter to set. The value is usually set experimentally, but the histogram of brightness in image can help to decide about the value. If the histogram is bimodal, the threshold can be chosen as the minimum value between two histogram "hills" (Fig. 1 left). If the histogram is not bimodal, more thresholds can be used to segment the objects (Fig. 1 right). Then, the equation is of the form

$$g(x,y) = \begin{cases} a, & f(x,y) > T_2 \\ b, & f(x,y) \leq T_2 \wedge f(x,y) > T_1 \\ c, & f(x,y) \leq T_1 \end{cases} \tag{2}$$

where a, b, c are the chosen output brightness values.

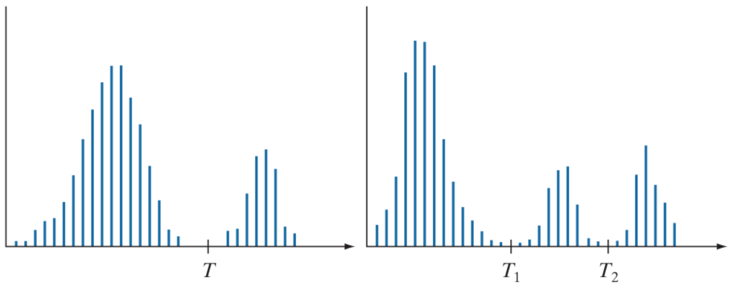

Figure 1: The example of bimodal histogram with one value T (left), and not-bimodal histogram where two thresholds T_1 and T_2 are used to segment the image (right).

Indexing

If we have a binary image that contains several segmented object areas (see Fig. 3), we can apply the indexing of objects. This step assign a unique number to each of objects, i.e. each pixel belonging to one of the areas will have this unique number. After indexing, we know the number of segments, the number of pixels in each segment, and also the width and height of segments. This information can be used for computing object features, which will be goal in next exercise.

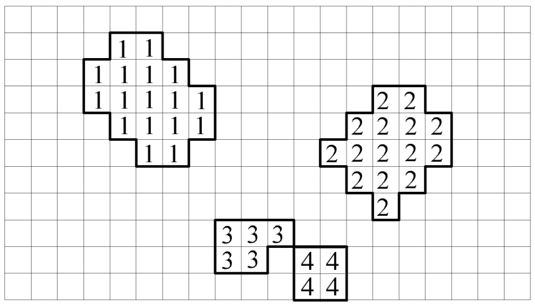

Figure 2: The result of indexing, each object has its own index value.

The flood fill algorithm is a typical algorithm for object indexing in images. In this algorithm, we go through all the pixels and if it belongs to the objects, we label this pixel with chosen constant and recursively flood fill four neighbours with this constant too. As one might expect, we label each object with different constant. The result of indexing can be seen in Fig. 2. We can also recursively flood fill eight neighbours, in this case, in Fig. 2 there would not be object with index 4 but it would be part of object 3.

TASKS

Load the prepared image, threshold it and index the objects. As the indexes, you can use the values 1,2,3, ..., but if you show the image, you should use higher values such that it will be visible - you can use a random color as it is shown in Fig. 3. We will use this image in next exercise for computing the image features.

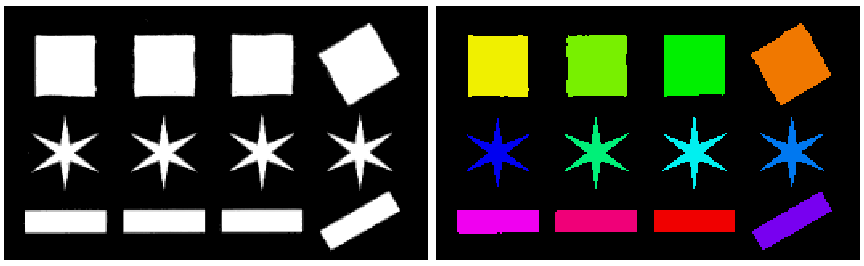

Figure 3: The input image containg several objects, and the indexed image.