Lab6: Morphological Segmentation: Watershed Algorithm.

Course 2021-2022

1 Introduction

In this practice we use advanced mathematical morphology concepts applied to the problem of **extracting the most important objects of an image**, also known as **image segmentation**.

One of the simplest techniques for segmentation is the *watershed*. In hydrology this word means the topographical boundary dividing two adjacent drainage basin, such as a ridge or a crest. A drainage basin (follow the link for more details) is the geographical region in which all water drains (for example, when it rains) to the same river.

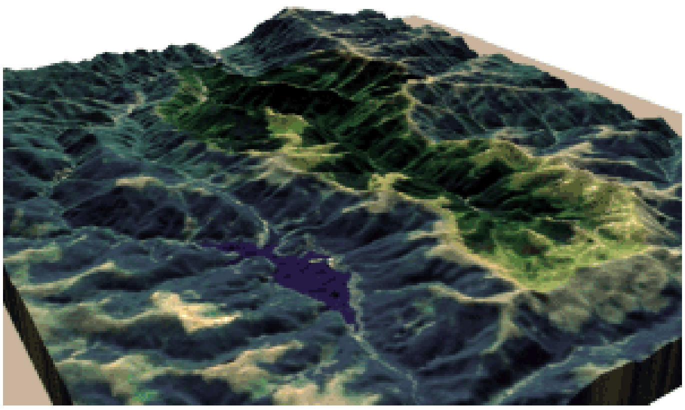


Figure 1. Example of the river basin.

As we know, in mathematical morphology a grayscale image is interpreted as a topographic relief. The watershed segmentation algorithm (follow the link for more details) extracts the watershed of the image through a process of flooding. In particular, the landscape is gradually flooded and when the water from two valleys contact (water overflows a basin), the algorithm places a barrier or dam to prevent water redistribution (mixing). The output image is formed by the different basins (each with a different label) and the set of dams or edges that delimit the basins.

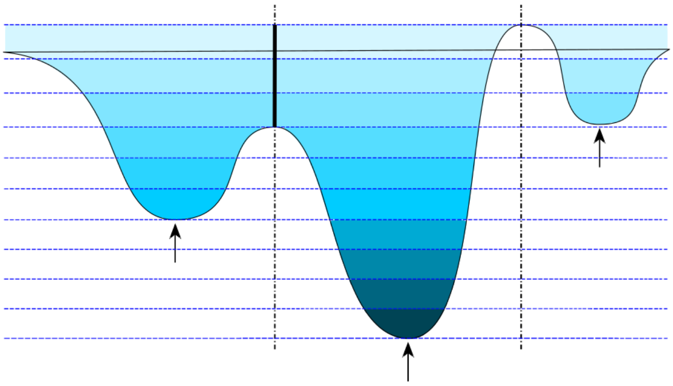


Figure 2. Flooding process and creation of dams (2D view of a section).

For practice you are going to use *watershed2.m* function (attatched in the zip lab file).

```
function L=watershed2(A, conn)
```

A direct application of the watershed algorithm are DEM images. These digital images give direct information about the height or elevation of a particular region. Generally, they are made from a satellite or an airplane. In these images, the intensity level of each pixel is directly related to the height above sea level of that part of the image.

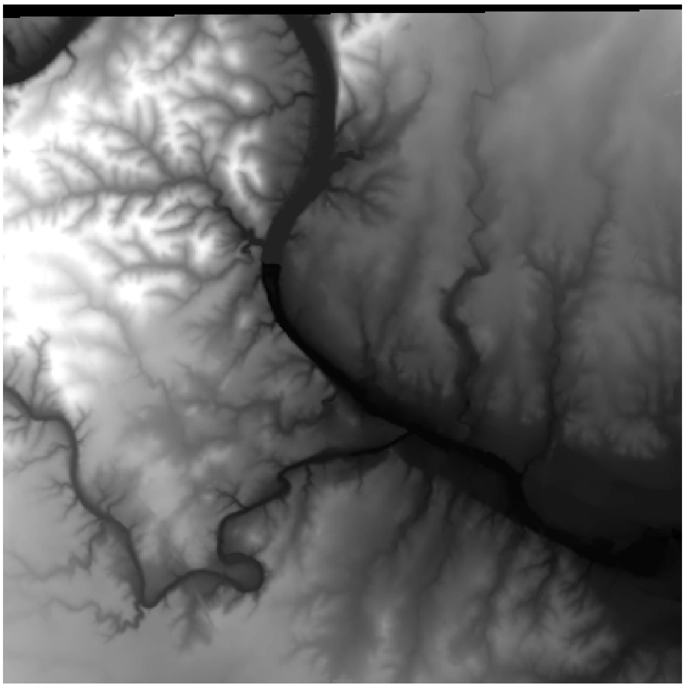
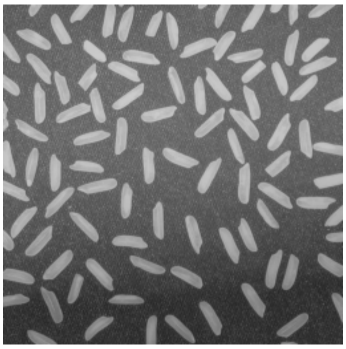


Figure 3. Example of an image of terrain elevation or DEM.

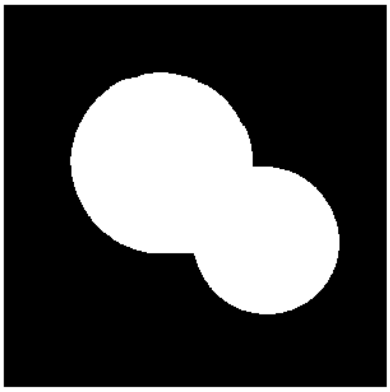
The real watershed of land may be of interest, for example, to measure the flow of a river to prevent flooding or to choose the location of a dam. In this particular case, we can use directly the watershed algorithm to extract them.

2 Extraction of objects of interest or segmenting a single image

To segment a gray level image where objects of interest are well contrasted with respect to the background and separated between them, we simply binarize image with a suitable threshold, filter it with a morphological operator and count the number of connected components (groups of neighboring pixels of the same value).




The same strategy can not be used when objects of interest touch each other:



In such situations, the watershed algorithm (applied to a proper transformation of an image and not directly to it) will be useful.
The steps to be followed in order to segment several distinctive objects in a given picture using the watershed algorithm are the following ones:

1. Binarize the image;
2. Apply a morphological filter to remove holes within the objects of interest;
3. Calculate the edges of objects (for example, use any of the morphological gradients we have seen in lab 4)
4. Calculate the distance from each pixel to the nearest edge (you can use the *bwdist* function). The resulting image, where the value of each pixel is the distance to the nearest boundary, is called the distance function (follow the link for more details);
5. Apply a small dilation (5x5) to the distance function to join maximas that are very close together (and thus avoid generating two different basins when applying watershed);
6. Apply watershed to the inverted distance function image;
7. Restrict the result of the watershed only to the objects of interest.

Exercise 1.

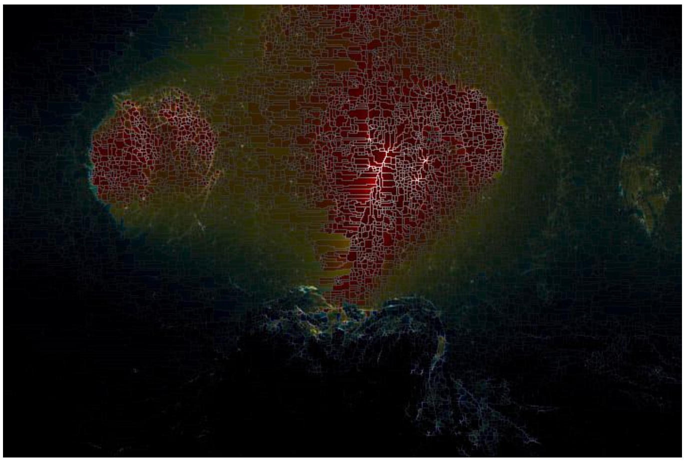
Exercise 1a.

Imagine you are one of the main characters of Stranger Things and you are trapped in a tunnel like the one seen in Figure 4 without visibility (the image is enhanced). However, you are lucky enough to know about Image Processing and the Watershed algorithm.

Figure 4. Stranger Things tunnel.

Extract watersheds from the image *Stranger.jpg* using the *watershed* function in order to see in which direction should you go.

Note: To visualize the obtained segmentation (an image where each pixel has a corresponding label of the basin to which it belongs), you can use the *label2rgb* function, that converts a label image into an RGB image with a different color for each label. In addition, to draw the labels over the original image, you can multiply each component (R, G and B) by the original image normalized to the range from 0 to 1. The result should be similar to that shown below.



Exercise 1b.

Perhaps you have noticed a large number of basins in the previous exercise. The explanation is that the *watershed* algorithm produces a basin for each minimum or valley of the image. This makes it very sensitive to noise and, in practice, we need to filter the image before applying it.

Use the same code as before in the exercise 1, but apply a morphological filter with a structuring element 10x10 to remove all maximas and minimas before segmenting the image. See any difference? Do the same with a structuring element of 20x20. From the point of view of the watershed, how to interpret the results?

Exercise 2.

Implement a function that, given an image of a certain number of coins on a white surface (which can touch each other or not), does the segmentation of each coin and counts the total amount of coins. Test your coin counter on the images *coins1.jpg*, *coins2.jpg*, *coins5.jpg* and discuss the results.
Try the other images of coins and discuss the results you get. What problems has the algorithm? Why? Do you have any suggestions how to fix it?