Zurich University
of Applied Sciences

School of
Engineering
ISC Institute of Signal Processing
and Wireless Communications

M. Weisenhorn, J. Rosset

8. November 2022

Laboratory – Watershed, Seed-Points, Labeling

1 Introduction

The goal of this laboratory is to get familar with the watershed algorithm. For this you will start from simple examples that can be found on the web and extend them so that the image `yeast.tif` will be automatically and semantically segmented. An approximate solution is sufficient, an exact solution can in general not be achieved.

2 Tasks

What follows is a list of individual program steps. After each step, visualize the outcome! You can use the functions proposed in section 4.

2.1 Thresholding

Generate a binarized image from `coins.tif` by thresholding the image with Otsu's algorithm. As a starting point you can use Thresholding-example. Explore the histograms and thresholds for the images `coins.tif`, `thGonz.tif`.

2.2 Seed points

In order to avoid oversegmentation, the watershed algorithm can be initialized with seed-points at the centers of the objects to be segmented. A starting point is provided in scikit-watershed-example.

Find seed points:

a) Apply the distance transform `scipy.ndimage.distance_transform_edt()` to the binary image obtained by applying Otsu's method.

b) Find local maxima by applying `skimage.feature.peak_local_max()`

c) The local maxima will serve as seedpoints. In this lab you should represent them by a matrix. First initialize a matrix with all zeros and with the same dimension as the image, by using the method `np.zeros_like()`. At those coordinates where the local maxima have been found write a `1` into the matrix. Finally, give individual integer values to the seed points, they will be used by the watershed algorithm to label the identified objects. For this, use the function `scipy.ndimage.label()`.

3 Watershed algorithm

a) Apply the wathershed function `skimage.segmentation.watershed()` to the negative of the distance image, or the negative of the gray value image, you can try out both.

b) Mask the resulting regions so that the segmented regions cover only the objects.

4 Visualization

Visualize the intermediate steps

a) Show the intermediate steps by e.g. the folowing lines of code

```
fig, axes = plt.subplots(ncols=4, figsize=(12, 3), sharex=True, sharey=
  True)
ax = axes.ravel()

ax[0].imshow(-image, cmap=plt.cm.gray, interpolation='none')
ax[0].set_title('Original Image')
ax[1].imshow(imageBinary, cmap=plt.cm.gray, interpolation='none')
ax[1].set_title('Otsu: image_binary')
ax[2].imshow(imageDistance/np.amax(imageDistance), cmap=plt.cm.jet,
  interpolation='none')
ax[3].scatter(peakCoords[:, 1].reshape((-1,)), peakCoords[:, 0].reshape
  ((-1,)), s=10, marker='o')
ax[4].imshow(labels, cmap=plt.cm.nipy_spectral, interpolation='none')
ax[4].set_title('Separated objects')

for a in ax:
        a.set_axis_off()

fig.tight_layout()
plt.show()
```