School of Computer Science 10th March, 2022
CMP9135M, Computer Vision

Workshop 3: Morphological Operation and Image Segmentation

The files required for this workshop can be downloaded from Blackboard (workshop3.zip)

Task 1: Basic Morphological Operations

Load binary test images (e.g. broken-text.tif , noisy-fingerprint.tif, coins.gif) into MATLAB with the usual commands, e.g.,

>> b = imread('noisy-fingerprint.tif'); The IPT function **strel** constructs structuring elements with a variety of shapes and size. Its basic syntax is:

```
>> se = strel(shape, parameters);
```

where shape is a string specifying the desired shape, and parameters is a list of parameters that specify information about the element's shape, such as its size. For example,

```
>> se = strel('disk', 2)
```

returns a disk-shaped structuring element with a radius of 2 pixels.

Using structuring elements of various shapes and sizes, try out the following MATLAB commands for morphological operations on some of the binary test images (please refer to help for further information):

- imdilate– image dilation
- imerode– image erosion
- imopen– image opening
- imclose– image closing

Task 2: Separation of touching coins

Load binary image (coins.gif), please use morphology operations to separate all the touching objects. You need to define a suitable structuring element for this task.

Task 3: Mushroom Image segmentation

Given two mushroom images (mushroom img1 and img2), your task is to segment mushrooms in each image using image segmentation techniques and morphology operations.

Task 4: Image segmentation using k-means

Given an image of your choice, your task is to segment the image using K-means. As discussed in the lecture, the algorithm is:

1) Given the number of clusters, k, initialise their centres to some values.
2) Go over the pixels in the image and assign each one to its closest cluster according to its distance to the centre of the cluster.
3) Update the cluster centres to be the average of the pixels added.
4) Repeat steps 2) and 3) until the cluster centres do not get updated anymore.
5) You can then do the post-processing by applying morphology operators.

Matlab has a built-in kmeans function. Please have a try.

You are also encouraged to implement your own k-means segmentation algorithm, such as,

```
function [centres, L] = my_kmeans (I, k)
```

I: the image to be segmented (greyscale to begin with)
k: the number of clusters (use a simple image with k=2 to begin with)
centres: a vector that contains the final cluster centres
L: an array the same size as the input image that contains the label for each of the image pixels, according to which cluster it belongs.

https://uk.mathworks.com/matlabcentral/fileexchange/8379-kmeans-image-segmentation