Computer Vision 1: Exercise Sheet 9

Summary:

1. Segmentation by k-means

1 Segmentation by k-means

We apply k-means segmentation on the grayscale version of `Elbphilharmonie.jpg` found on Moodle. The k-means algorithm can be applied on grayscale images to cluster them according to their grayscale value.

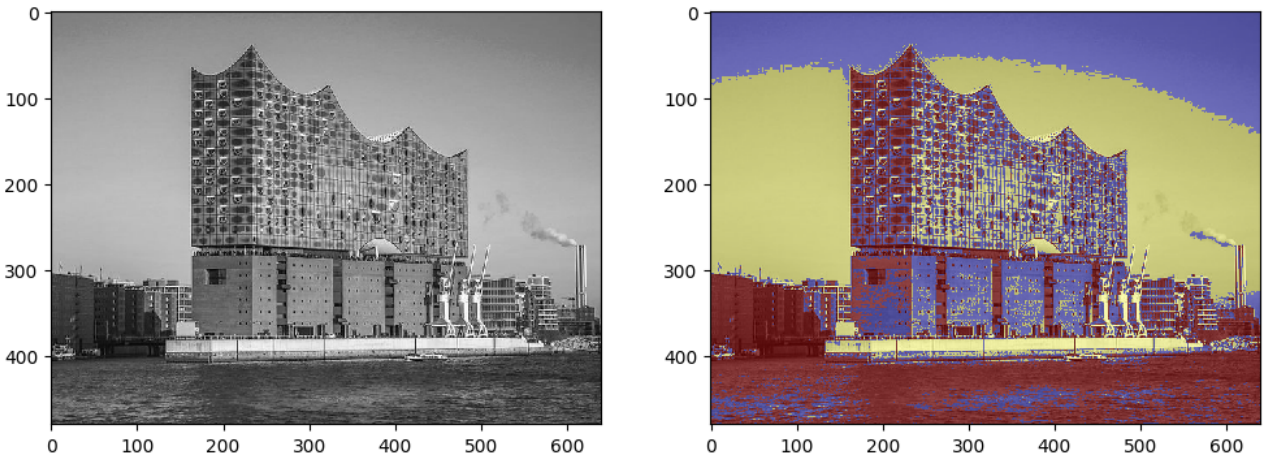

Figure 1: Original grayscale image (left) and a k-means clustering result with $k = 3$ (right). The colored regions indicate the segments.

Convert the image to grayscale. Then, implement the k-means algorithm for clustering the image using NumPy. Follow the steps in the list below:

1. Randomly choose k values among all grayscale values in the image as initial cluster centers.
2. For each point p_j in the image, calculate which of the cluster centers c_i, $i = 1, \ldots, k$ it belongs to. The point belongs to the cluster to whose center is closest to it, i.e. the cluster c_i that minimizes $|p_j - c_i|$. Let us call this cluster assignment by $c(p_j) \in \{1, 2, \ldots, k\}$.

   Note that p_j and c_i refer to grayscale values, not image coordinates.
3. Now we know for each point p_j the cluster assignment $c(p_j)$. Use it to update the cluster centers. For cluster i, the new cluster center is the average of grayscale values of all points for which $c(p_j) = i$.
4. If there was a change in the cluster centers, repeat from step 2. To determine if there was a change, use a small value such as $\epsilon = 0.001$ against which you compare the absolute value of the difference of each new cluster center and old cluster center.

Visualize the clustering result. Use `skimage.color.label2rgb` with argument `kind='overlay'` to create a colored label image. An example is shown in Figure 1.