Texture Segmentation Using K-means and Gabor Filters

April 15, 2009

The goal of the lab is to segment textured images using a bank of Gabor filters and the k-means clustering algorithm.

1 K-means algorithm

The k-means is a data clustering (classification) algorithm. It allows to subdivide a group of data points in k groups, where the number k is an input (provided by the user). The basic idea behind the algorithm is to define k centers, with each center being the barycenter of each group. K-means consists of the following steps :

1. Start by Partitioning the data in k groups (randomly) and compute the mean of each group. The computed means are the initial values of the centers. Another initialization strategy consists of selecting k points chosen randomly as centers.
2. Assign each data point to the group of the closest center to the considered point.
3. Recompute the new positions of the centers using the assignments obtained in the previous step.
4. Repeat steps 2 and 3 until convergence, which means until all the points do not change affectation (or equivalently the centers are stable).

Note that the k-means algorithm is equivalent to minimizing the following functional :

$$V = \sum_{i=1}^{k} \sum_{x_j \in G_i} \|x_j - \mu_i\|^2$$

where x_j is a data point, G_i is the group (cluster) i and μ_i its center. The two steps of the k-means amount to an alternate minimization of the energy V over the centers μ_i and the assignments G_i.

2 A texture segmentation algorithm

Given an image composed of different textures, the aim is to separate the different components, that is to group points in the image if they have similar texture content.

We proceed as follows :

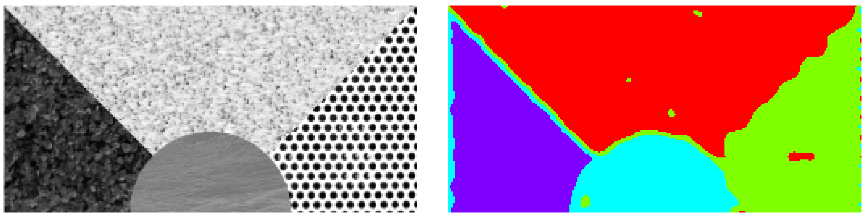

FIG. 1 – A textured image and its segmentation in 4 regions using k-means and a Gabor filter bank

1. At each point of the input image, we compute the response of a bank of Gabor filters. This means that we use multiple frequencies and orientations. We store the different responses in a *feature vector*.
2. The obtained feature vectors are grouped using k-means. The different clusters provide a segmentation of the input image.

Note that in practice we will consider an initial spatial bandwidth σ and its corresponding frequency f_0, then select multiples of σ (while updating the frequency f_c) to build the filter bank. We also consider a number of orientations and sample them uniformly in $[0\ \pi]$.

3 Programming assignment

Here are the programming tasks :

1. Complete the function `kMeans` which takes as input a list of data points `points`, an array `pointsAssignment` which contains the group of each point, the list of centers `centers` and the number of centers (groups) `centerN` . This function should fill in `pointsAssignment` and `centers` using the k-means algorithm. Note that the function already checks the sizes of the inputs and reallocates the arrays if the sizes are incorrect.
2. Complete the `main` function. In particular, fill in the array `filtered` such that `filtered[i][j]` (which is an image, i.e. `CImg<float>`) contains the image filtering response for the i-th frequency and j-th orientation. This is done using the function `GaborFilter`. You will also declare and fill in `features` (which is a `CImgList<float>`) with the vector of features from the array `filtered`.

We provide a test image `multitexture.png`. The segmentation result is saved on the disk (`kmeansGaborSegmentation.png`).