Albert-Ludwigs-Universität Freiburg
Lecture: Mobile robotics II
Winter term 2011/2012

Institut für Informatik
Exercise: Prof. Dr. Maren Bennewitz
Lab instructor: Dipl.-Phys. Jonas Rist

Exercise: Clustering and Bag of Words

Introduction
In this octave exercise you will first implement the k-means and GMM (Gaussian Mixture Model) clustering algorithms. Then you will create a BOW (Bag of Words) representation of sample images and use it to recognize object classes in these images.

K-means Clustering
Complete the function *kmeans.m* by implementing the necessary steps for k-means clustering. The result should be an octave function with the signature

```
[centroids, labels, compactness] = kmeans(data, number_of_clusters)
```

Consult the comments in `kmeans.m` for the definition of the input and output variables.

1. **Initialization** The initialization of the centroid vector has a strong influence on the performance of the algorithm. A reasonable and simple possibility is to sample the initial centroids from the input data randomly (take care not to sample the same data vector twice!). An even better way might be to spread the centroids evenly over the input data (only if there are not too many outliers!).

2. **Testing** Test your implementation on synthetic data first. Use the octave data files in the `/data` folder, e.g. `5_gaussian_clusters.dat`. Generate Gaussian clusters on your own for testing (you can use the function `generate_test_data_2d.m`, see the code for how to use it).

GMM Clustering
Complete the function `gmm.m` by implementing the necessary steps for estimating a GMM. The result should be an octave function with the signature

```
[centroids, covar, labels] = gmm(data, number_of_clusters)
```

Consult the comments in `gmm.m` for the definition of the input and output variables. Perform the same steps as for k-means. Initialize the GMM with some iterations of k-means to achieve faster convergence.

Bag of Words
The folder `/data/bow-object-recognition` contains sample images of toys. The `data.info` file lists image names and corresponding object classes and ids.

1. **SURF Features** A frequent approach to feature extraction from images is SURF (Speeded Up Robust Features). For each image file in `/data/bow-object-recognition`, there is a `.features`-file containing surf features extracted from that image. The first column is a class id, and the next 64 columns contain the 64 components of a SURF descriptor.

2. **Codebook** You can use the function `create_codebook.m` to create a codebook with k-means. Note: It will only work if you have implemented the function `kmeans.m` correctly.

3. **Histograms** Write a function that creates histograms of single images by assigning the image features to clusters from the codebook and counting their frequency.

4. **Object recognition** Choose two distinct subsets from the images. Create a codebook from one set (the *training set*). Create histograms of the other set (the *test set*) and try to classify the images by matching their histograms against histograms from the training set. E.g, a simple strategy for the classification of a test image is:

   (a) Compute the euclidean distance between the histogram of the test image and the histograms of all training images

   (b) Average the distances over each class of training images

   (c) Assign the class with the lowest average distance to the test image

   Begin with small trainig and test sets!