Bag of Word: Place Recognition

Background

We will use the bag-of-words (also known as bag-of-features, bag-of-visterms) model for object recognition. First of all, check the bag-of-words tutorial at http://people.csail.mit.edu/torralba/shortCourseRLOC/index.html

Include the sift package in Matlab's path: addpath('../sift');
- Have a look at the documentation

- In the remaining of the exercise, we will use images from the KTH-IDOL2 database, containing a sequence of images recorded by a robot driving through an office building. In exercise 1 of this summer school you downloaded two sequences from the database. We will work with those. Get the sequence information with: sequenceInfo = getIDOLSequenceInfo('min_cloudy1');

In this assignment, we will use bags-of-features to represent and recognize the place where the robot is. We will use the first sequence ('min_cloudy1') to learn the places and the second sequence ('min_cloudy2') to test the recognition.

For the training phase, the first thing to do is to find the prototypical visual features, that is, the visual words in the dictionary that we will use for recognition. To do that, we take a large number of interest points found in the first sequence and cluster these into K clusters. This will give the K visual words. Once we have the words, we can make a signature for every image in the sequence. The signature is a frequency histogram of the visual words. So for every image, we need to count how often each of the K words appear in that image. We can then store the signatures of every image along with the place label (sequenceInfo(i).placeID).

With all the signatures and place labels stored, we can try to recognize the place in the images in the second sequence. We use the same K words as in the training phase, and make a signature for the given images. By comparing that signature to all the learned signatures, we can find the nearest neighbor. This will give us the place label.

Calculate Features

We have pre-calculated all the SIFT interest points and detectors for the images in the database. There is a pointer to the file containing the SIFT descriptors in sequenceInfo(i).siftFileName. If you load the sift file, it will give you a matrix containing all the interest points found in the image. Every row of the matrix contains an interest point. The first columns give the xpos, ypos, scale, and orientation, followed by the 128 columns of the sift descriptor.
- Make a large matrix where you store all the descriptors found in the sequence (excluding the xy pos and scale and orientation). Each descriptor should be stored in a row of the matrix. To prevent a too large matrix, use only every 10th image in the sequence.

Cluster Features

After you have all the descriptors, use k-mean clustering to cluster our descriptors. Use MATLAB's kmeans function to cluster descriptors using different k values. Typically, a large number of clusters (e.g., 500, 1000, 2000) has been used in the literature. You need to experiment with the number of clusters with respect to the number of local features obtained in the training data.

Create the bag-of-words histograms (or signatures)

We need to map all the raw SIFT descriptor in an image to its visual word:
Each raw descriptor is assigned to the word vector (cluster center) it is nearest to in terms of Euclidean distance. [see provided `dist2.m` code for fast distance computations; note that `[minvals, mininds] = min(D, [], 2);` returns a vector containing the minimum value per row of the matrix D along with the column indices where each of the mins are found.]

The histogram which we will call the ‘signature’ for image I_j is a k-dimensional vector: $F(I_j) = [freq_{1,j}, freq_{2,j}, \ldots, freq_{k,j}]$, where each entry $freq_{i,j}$ counts the number of occurrences of the i-th visual word in that image, and k is the number of total words in the vocabulary. In other words, a single image’s list of n SIFT descriptors yields a k-dimensional bag-of-words histogram. [Matlab’s `histc` is a useful function].
Create a signature (histogram) for every image in the first sequence. Label every histogram with the label sequenceInfo(i).placeID so that you know which room the signature corresponds to.

Testing

Now go back and load the features for the other sequence getIDOLSequenceInfo(‘min_cloudy2’); This new set has not been trained on so it can be used for testing the technique for similarity. Now step through each image and using the same set of visual words computed on the previous sequence compute a histogram signatures for an image and measure that histograms distance to all the signatures computed from the previous step. Finding the minimum distance signature and its associated placeID gives you your classification for this new image. Compare that and the actual placeID from the structure to check for successful classification. Do this for all the images in the second sequence and calculate the success rate.