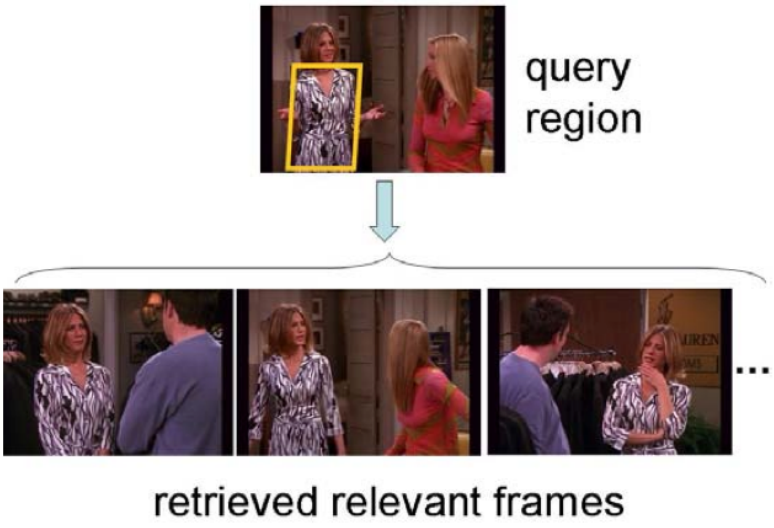
Assignment 3

Computer Vision 2014-2015
University of Bern

1 Video search with bags of visual words (Due on 09/12/2014)

For this assignment (credit Kristen Grauman[1]) , you will implement a video search method to retrieve relevant frames from a video based on the features in a query region selected from some frame (Fig. 1);



[1] http://www.cs.utexas.edu/ grauman/

1.1 Data [25 points]

First you need to provide some video data of your own which you can capture with your own camera or download from the web. To compute the sift features visit: 'http://www.vlfeat.org'. For each image, save a '.mat' file that contains the following variables, where n is the number of detected SIFT features in that image:

- descriptors: $n \times 128$ double // the SIFT vectors as rows
- imname: 1×57 char // name of the image file that goes with this data
- numfeats: 1×1 double // number of detected features
- orients: $n \times 1$ double // the orientations of the patches
- positions: $n \times 2$ double // the positions of the patch centers
- scales: $n \times 1$ double // the scales of the patches

1.2 Provided code

We are providing some starter code for this assignment:

- `loadDataExample.m`

  Run this first and make sure you understand the data format. It is a script that shows a loop of data files, and how to access each descriptor. It also shows how to use some of the other functions below.

- `displaySIFTPatches.m`

  Given SIFT descriptor info, it draws the patches on top of an image.

- `getPatchFromSIFTParameters.m`

  Given SIFT descriptor info, it extracts the image patch itself and returns as a single image.

- `selectRegion.m`

  Given an image and list of feature positions, it allows a user to draw a polygon showing a region of interest, and then returns the indices within the list of positions that fell within the polygon.

- `dist2.m`

  A fast implementation of computing pairwise distances between two matrices for which each row is a data point.

- `kmeansML.m`

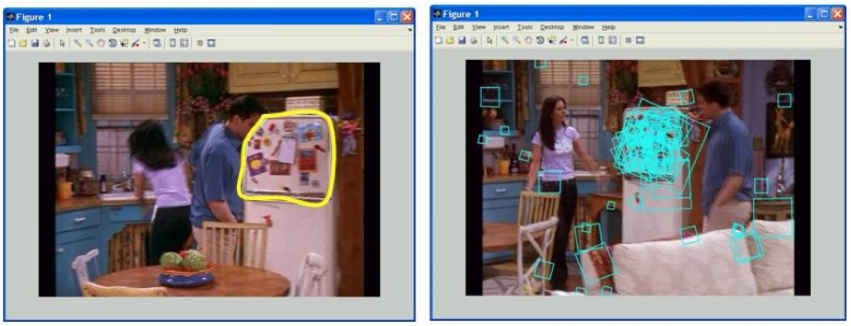
  A faster k-means implementation that takes the data points as columns.

You are not required to use any of these functions, but you will probably find them helpful.

1.3 What to implement and discuss in the write-up

Write one script for each of the following (along with any helper functions you find useful), and in your pdf writeup report on the results, explain, and show images where appropriate.

1. **Raw descriptor matching [15 points]:** Allow a user to select a region of interest (see provided **selectRegion**) in one frame, and then match descriptors in that region to descriptors in the second image based on Euclidean distance in SIFT space. Display the selected region of interest in the first image (a polygon), and the matched features in the second image, something like in the following figure. Use the two images and associated features in the provided file **twoFrameData.mat** (in the gzip file) to demonstrate. Note, no visual vocabulary should be used for this one. Name your script **rawDescriptorMatches.m**.



2. **Visualizing the vocabulary [20 pts]:** Build a visual vocabulary. Display example image patches associated with two of the visual words. Choose two words that are distinct to illustrate what the different words are capturing, and display enough patch examples so the word content is evident (e.g., say 25 patches per word displayed). See provided helper function **getPatchFromSIFTParameters**. Explain what you see. Name your script **visualizeVocabulary.m**

3. **Full frame queries [20 pts]:** After testing your code for bag-of-words visual search, choose 3 different frames from the entire video dataset to serve as queries. Display the $M = 5$ most similar frames to each of these queries (in rank order) based on the normalized scalar product between their bag of words histograms. Explain the results. Name your script **fullFrameQueries.m**

4. **Region queries [20 pts]:** Select your favorite query regions from within 4 frames (which may be different than those used above) to demonstrate the retrieved frames when only a portion of the SIFT descriptors are used to form a bag of words. Try to include example(s) where the same object is found in the most similar M frames but amidst different objects or backgrounds, and also include a failure case. Explain the results, including possible reasons for the failure cases. Name your script **regionQueries.m**

1.4 Tips: overview of framework requirements

The basic framework will require these components:

- **Form a visual vocabulary.** Cluster a large, representative random sample of SIFT descriptors from some portion of the frames using k-means. Let the k centers be the visual words. The value of k is a free parameter; for this data something like $k = 1500$ should work, but feel free to play with this parameter [see Matlab's **kmeans** function, or provided **kmeansML.m** code]
- **Map a raw SIFT descriptor to its visual word.** The raw descriptor is assigned to the nearest visual word. [see provided **dist2.m** code for fast distance computations]
- **Map an image's features into its bag-of-words histogram.** The histogram for image I_j is a k-dimensional vector: $F(I_j) = [\mathbf{freq}_{1,j}, \mathbf{freq}_{2,j}, ... \mathbf{freq}_{k,1}]$, where each entry $\mathbf{freq}_{i,j}$ 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]
- **Compute similarity scores.** Compare two bag-of-words histograms using the normalized scalar product.

- **Sort the similarity scores** between a query histogram and the histograms associated with the rest of the images in the video. Pull up the images associated with the M most similar examples[see Matlab's **sort** function].
- **Form a query from a region within a frame.** Select a polygonal region interactively with the mouse, and compute a bag of words histogram from only the SIFT descriptors that fall within that region. Weight it with tf-idf. [see provided **selectRegion.m** code].
- **Compute nearest raw SIFT descriptors.** Use the Euclidean distance between SIFT descriptors to determine which are nearest among two images' descriptors. That is, "match" features from one image to the other, without quantizing to visual words.

1.5 What to send via ILIAS

(a) Your commented Matlab code (zip file),

(b) A pdf file containing your name, results, figures, comments and a brief explanation of your implementation strategy.