Part I

1. By taking any positions that are local maxima in a scale-space, filter a would yield scale-invariant interest points. On the contrary, filter b takes any positions whose filter response exceeds a threshold, so it does not take scale into account. Thus, filter b would not yield as repeatable results as filter a.

Nevertheless, filter a and filter b are both rotation-invariant. They capture distinct interest points, such as corners where the gradient magnitude is large. Thus filter a and filter b both produce distinct interest points.

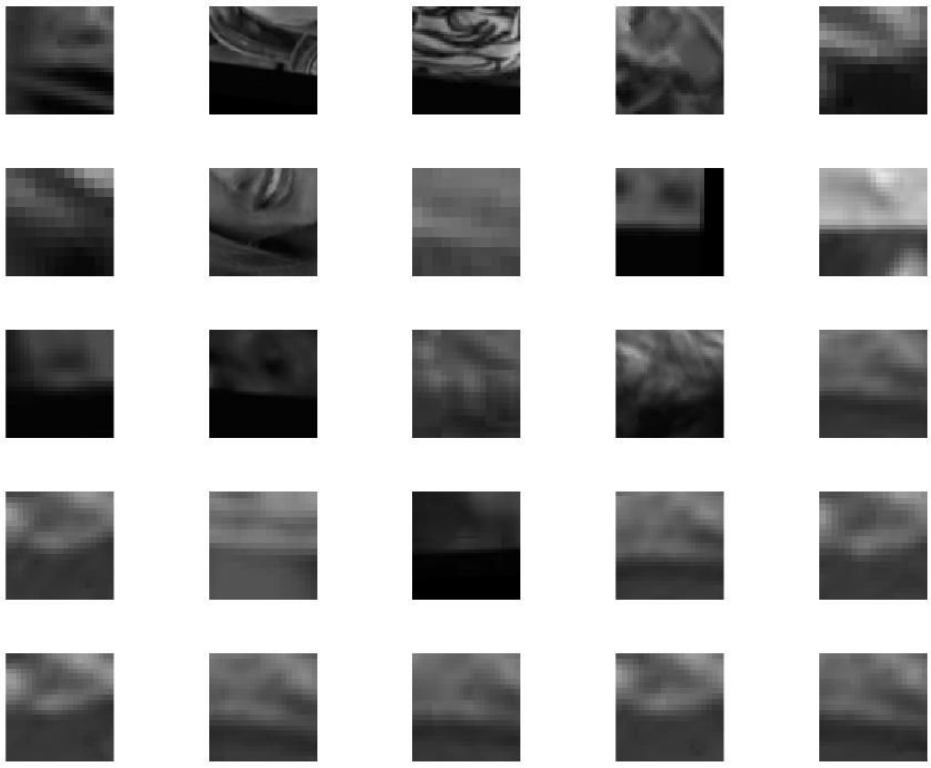
2.

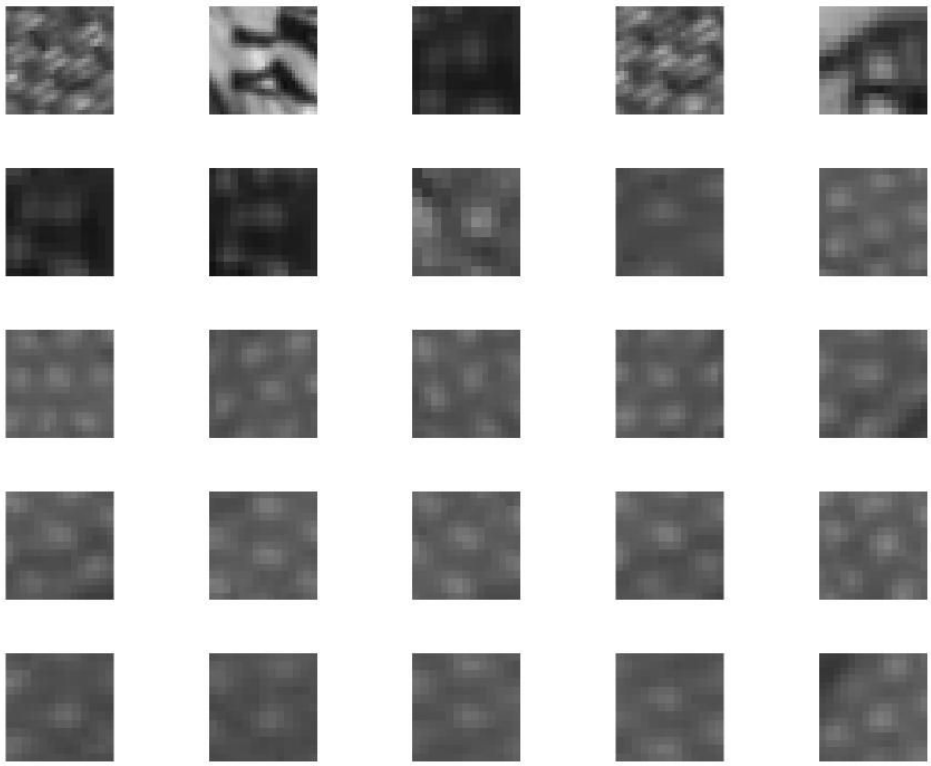
Each value recorded in a single dimension of a SIFT keypoint descriptor signifies the magnitude of gradient orientations in a sub-patch of a descriptor. SIFT descriptor uses histogram to bin pixels within each subpatch according to their orientation.

Part II

2.

Top 25 patches for each visual word





The script visualize_vocabulary.m. has a helper function build_visual_vocab.m.
25 patches are displayed for each of the two visual words chosen. The first one has a lighter color on the upper half and a dark one below. The second visual word has net-like pattern with intertwining dark and light colors. These patches are very low-level compared to the images provided, nevertheless, they display distinct structure.

3.
Execution might take up to 5 mins.
If you want to have a bag-of-words consisting of the full range of images, change 400 to len_fnames in range1 = randperm(len_fnames,400). I chose 400 instead of using the whole image set because of computational cost.

fullFrameQueries.m has a helper function BuildBag.m, which convert a chosen range of frames (400 random frames) into histograms of bag of words.
The figures below are the results of the three queries.

Query Image

Match Rank 1

Match Rank 2

Match Rank 3

Match Rank 4

Match Rank 5

This **first query image** has a couple sitting at a table. The first and fourth retrieved results are from the same set. It's interesting to see that the fourth ranked picture is closer to the query image than the second and third. The rest all have people/person inside the frame. One reason
is the limited number of images we use to build bag of visual vocabs.

Query Image

Match Rank 1

Match Rank 2

Match Rank 3

Match Rank 4

Match Rank 5

The retrieval result of **the second query** is not as good as the first. Nevertheless, all the retrievals have people sitting in the middle of the frame.

Query Image

Match Rank 1

Match Rank 2

Match Rank 3

Match Rank 4

Match Rank 5

The retrieval result of the **third query** is mixed. None of them closely resembles the query image. Interestingly, the Rank 3 image and Rank 5 image are from the same set, which demonstrates the scale-invariant property of the SIFT.

4.

The first query image is able to retrieve frames that show the same couple in the same scene three out of five times. Both the man and woman are the major focus of the descriptors for the image. The script retrieves 3 frames that are nearly identical to the original query image. However, the Rank 2 frame is very different from the query image, the reason might be that only the woman in the query image is taken into account.

The retrieval of second query image has high success rate . The Rank 1, 2 image are from the same scene as query image. However, the Rank 3,4,5 image are all different from the query image. I think it might be the case that the range of images we are trying to retrieve from only contain so many that are close to the query image.

Query Image

Match Rank 1

Match Rank 2

Match Rank 3

Match Rank 4

Match Rank 5

The third query is a failure case in the sense that none of the retrieved images look alike the query image. One reason might be that the wall pattern is too generic and ambiguous for the SIFT detector.

Query Image

Match Rank 1

Match Rank 2

Match Rank 3

Match Rank 4

Match Rank 5

The region of interest of t**he fourth query image** is not in the center of the scene but a local pattern. It also has a smaller scale. We retrieve dotted clothing patterns in Rank 1, Rank 2 and Rank 5 images, which shows that SIFT descriptors are scale and orientation invariant.

Query Image

Match Rank 1

Match Rank 2

Match Rank 3

Match Rank 4

Match Rank 5

5.

DeepFC Query Image

DeepFC Match Rank 1

DeepFC Match Rank 2

DeepFC Match Rank 3

DeepFC Match Rank 4

DeepFC Match Rank 5

DeepFC Match Rank 6

DeepFC Match Rank 7

DeepFC Match Rank 8

DeepFC Match Rank 9

DeepFC Match Rank 10

SIFT Query Image

SIFT Match Rank 1

SIFT Match Rank 2

SIFT Match Rank 3

SIFT Match Rank 4

SIFT Match Rank 5

SIFT Match Rank 6

SIFT Match Rank 7

SIFT Match Rank 8

SIFT Match Rank 9

SIFT Match Rank 10

DeepFC Query Image

DeepFC Match Rank 1

DeepFC Match Rank 2

DeepFC Match Rank 3

DeepFC Match Rank 4

DeepFC Match Rank 5

DeepFC Match Rank 6

DeepFC Match Rank 7

DeepFC Match Rank 8

DeepFC Match Rank 9

DeepFC Match Rank 10

SIFT Query Image

SIFT Match Rank 1

SIFT Match Rank 2

SIFT Match Rank 3

SIFT Match Rank 4

SIFT Match Rank 5

SIFT Match Rank 6

SIFT Match Rank 7

SIFT Match Rank 8

SIFT Match Rank 9

SIFT Match Rank 10

I created a 'BuildFC.m' helper function that builds a bag of vocab for feature deepFC7.

The retrieval results obtained using the deep convolutional neural network features are a lot better than SIFT bag-of-words because we are trying to detect scene-level features. For the first photo (woman in purple T-shirt), CNN does remarkably well. CNN captures global and high level information for an image better than SIFT. The retrieval results of the first image returned by SIFT capture local details but do not resemble the query image at the scene level. Concretely,SIFT is a 128 dimensional vector that summarizes 16×16 window patch. The SIFT is obtained by dividing the 16×16 window into 4×4 bins. Each bin has 8 orientation bins or channels.

SIFT does not learn the representation by itself; it is hard-coded.

Thus SIFT preserves low-level features but doesn't make use of hierarchical layer-wise representation learning while the CNN is a hierarchical deep learning model which is able to model data at scene-level.

Part III

I will get a bag-of-words per image. Then, I will combine them and get a histogram that capture the global frequency of visual words in all images. For words with top 10 highest weight, I will remove them from my histogram and fill them with 0. Then I will apply additional weighting to tf-idf with higher weights corresponding to important words.Due to time constraint, I was not able to implement.