VLFeat Library-based SIFT Demo

This script uses the VLFeat library to provide a demo of the SIFT feature and its use for matching. This script is used within the course "CS491/691: Autonomous Mobile Robot Design".

To use this script, you first have to install VLFeat: http://www.vlfeat.org/index.html

Author: Kostas Alexis

Part 1: Basic SIFT functionality

Subsequently, SIFT feature extraction, its use for matching and some of its properties will be visualized.

```matlab
pxf = fullfile(vl_root,'figures','demo');
randn('state',0);
rand('state',0);
figure(1); clf;

Load a Figure
```

```matlab
img = imread(fullfile(vl_root,'data','im_a.jpg'));

image(img); colormap gray;
axis equal; axis off; axis tight;
```
Convert to the required format

The `vl_sift` command requires a single precision gray scale image. It also expects the range to be normalized in the [0,255] interval (while this is not strictly required, the default values of some internal thresholds are tuned for this case). The image $I$ is converted in the appropriate format by

```matlab
I = single(rgb2gray(I)) ;
clf ; imagesc(I)
axis equal ; axis off ; axis tight ;
```
Run SIFT

The matrix $F$ has a column for each frame. A frame is a disk of center $F(1,2)$, scale $F(3)$ and orientation $F(4)$. We visualize a random selection of 50 features by:

```matlab
[f,d] = vl_sift(I);

hold on;
perm = randperm(size(f,2));
.sel = perm(1:50);
hi = vl_plotframe(f(:,sel)); set(h1,'color','k','linewidth',3);
h2 = vl_plotframe(f(:,sel)); set(h2,'color','y','linewidth',2);
```
We can also overlay the descriptors by

delete([h1 h2]);

h3 = vl_plotshiftdescriptor(d(:, sel), f(:, sel));
set(h3, 'color', 'k', 'linewidth', 2);

h4 = vl_plotshiftdescriptor(d(:, sel), f(:, sel));
set(h4, 'color', 'g', 'linewidth', 1);

h1 = vl_plotframe(f(:, sel)); set(h1, 'color', 'k', 'linewidth', 3);

h2 = vl_plotframe(f(:, sel)); set(h2, 'color', 'y', 'linewidth', 2);
clf ; imagesc(I(1:200,1:200,:))
axis equal ; axis off ; axis tight ;
hold on;

fc = [100;100;10;-pi/8] ;
[f,d] = vl_sift(I,'frames',fc);

h3 = vl_plotsiftdescriptor(d,f) ; set(h3,'color','k','linewidth',3);

h4 = vl_plotsiftdescriptor(d,f) ; set(h4,'color','g','linewidth',2);

h1 = vl_plotframe(f) ; set(h1,'color','k','linewidth',4);

h2 = vl_plotframe(f) ; set(h2,'color','y','linewidth',2);
Custom Keypoints with orientations

fc = [100;100;10;0] ;
[f,d] = vl_sift(I,'frames',fc,'orientations') ;
delete([h1 h2 h3 h4]) ;

h3 = vl_plotsiftdescriptor(d,f) ; set(h3,'color','k', 'linewidth',3);
h4 = vl_plotsiftdescriptor(d,f) ; set(h4,'color','g', 'linewidth',2);
h1 = vl_plotframe(f) ; set(h1,'color','k','linewidth',4) ;
h2 = vl_plotframe(f) ; set(h2,'color','y','linewidth',2) ;
Part 2: SIFT Basic Matching

SIFT descriptors are often used find similar regions in two images. `vl_ubcmatch` implements a basic matching algorithm. Let $I_a$ and $I_b$ be images of the same object or scene. We extract and match the descriptors by:

```matlab
pfx = fullfile(vl_root,'figures','demo');
randn('state',0);
rand('state',0);
figure(1);clf;
```

Create Image Pair

```matlab
Ia = imread(fullfile(vl_root,'data','im_a.jpg'));
```
Ib = imread(fullfile(vl_root,'data','im_b.jpg'));

Extract Features and Match

[fa,da] = vl_sift(im2single(rgb2gray(Ia)));
[fb,db] = vl_sift(im2single(rgb2gray(Ib)));

[matches, scores] = vl_ubcmatch(da,db);

drop, perm = sort(scores, 'descend');
matches = matches(:, perm);
scores = scores(perm);

figure(1); clf;
imagesc(cat(2, Ia, Ib));
axis image off;
vl_demo_print('sift_match_1', 1);

figure(2); clf;
imagesc(cat(2, Ia, Ib));

xa = fa(1,matches(1,:));
xb = fb(1,matches(2,:)) + size(Ia,2);
ya = fa(2,matches(1,:));
yb = fb(2,matches(2,:));

hold on;
h = line([xa; xb], [ya; yb]);
set(h,'linewidth', 1, 'color', 'b');
For each descriptor in da, `vl_ubcmatch` finds the closest descriptor in db (as measured by the L2 norm of the difference between them). The index of the original match and the closest descriptor is stored in each column of `matches` and the distance between the pair is stored in `scores`.

Matches also can be filtered for uniqueness by passing a third parameter to `vl_ubcmatch` which specifies a threshold. Here, the uniqueness of a pair is measured as the ratio of the distance between the best matching keypoint and the distance to the second best one (see `vl_ubcmatch` for further details).
Part 3: Detector parameters

The SIFT detector is controlled mainly by two parameters: the peak threshold and the (non) edge threshold.

The *peak threshold* filters peaks of the DoG scale space that are too small (in absolute value). For instance, consider a test image of 2D Gaussian blobs:

```matlab
randn('state',0);
rand('state',0);

I = double(rand(100,500) <= .005);
I = (ones(100,1) * linspace(0,1,500)) .* I;
I(:,1) = 0; I(:,end) = 0;
I(1,:) = 0; I(end,:) = 0;
I = 2*pi^4^2 * vl_imsmooth(I,4);
I = single(255 * I);

figure(1); clf;
imagesc(I); colormap gray;
axis equal; axis off; axis tight;
hold on;
```

We run the detector with peak threshold *peak_thresh* by setting the relevant parameters in the *vl_sift* the command. Similarly we run the detector with edge threshold *edge_thresh* by setting the relevant parameters in the *vl_sift* command. We tend to obtain more features as *edge_thresh* is increased.

```matlab
tpr = [0 10 20 30];
for tp=tpr
    f = vl_sift(I, ... 
         'PeakThresh', tp, ... 
         'EdgeThresh', 10000, ... 
         'FirstOctave', -1);

    h1 = vl_plotframe(f);
    set(h1,'color','k','linewidth',3);
    h2 = vl_plotframe(f);
    set(h2,'color','y','linewidth',2);

    vl_demo_print(sprintf('sift_peak_%d',find(tp==tpr)));
    pause(1)
    delete(h1);
    delete(h2);
end
```
Mosaic using SIFT - Demo

```python
mosaic = sift_mosaic_demo(Ia, Ib);
```

Warning: Refinement disabled as fminsearch was not found.
1067 tentative matches

617 (57.83%) inliner matches out of 10
Mosaic