CS-E4850 Computer Vision, Answers to Exercise Round 3

Yangzhe Kong, Student number: 765756

October 3, 2019

Exercise 1. Matching Harris corner points.

a) The code for implementation of normalized cross-correlation (NCC) can be found in **NCCSimilarity.m**

```
function result=NCCSimilarity(patchA,patchB)
mean_a=mean(mean(patchA));
mean_b=mean(mean(patchB));
patchA_nor=patchA-mean_a;
patchB_nor=patchB-mean_b;
result=sum(sum(patchA_nor.*patchB_nor))+
/sqrt(sum(sum(patchA_nor.^2))*sum(sum(patchB_nor.^2)));
end
```

b) The number of correct correspondences using SSD is only 20.
The number of correct correspondences using NCC instead of SSD is 295.
The best matches of these two measures are shown in Figure 1 and 2.

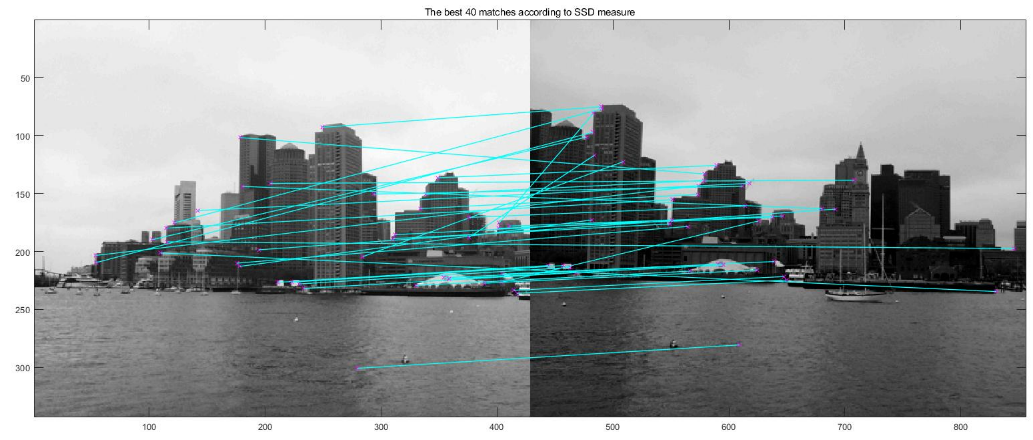

Figure 1: The best matches using SSD.

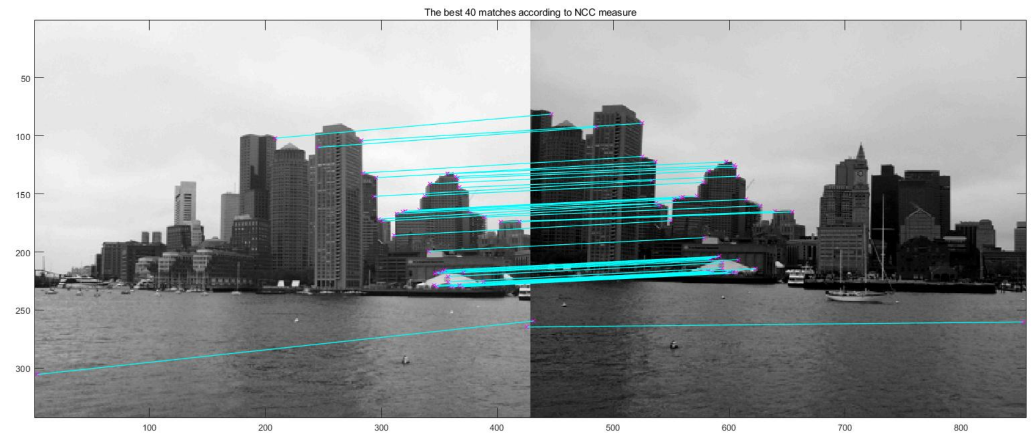

Figure 2: The best matches using NCC.

c) The NCC measure performs better. It's because SSD measure merely uses sum of squared differences as distance metrics. But NCC measure uses a more delicate way.

Normalization allows NCC to compare patches with different value ranges and see if there's any correlation between them. So NCC measure is more robust to image transformation.

Exercise 2. Matching SURF regions.

a) Only 2 of the 5 best matches are correct correspondences using nearest neighbor distance.
   And all of the 5 best matches are correct correspondences using nearest neighbor distance ratio (NNDR). The best 5 matches of these two measures are shown in Figure 3 and Figure 4.

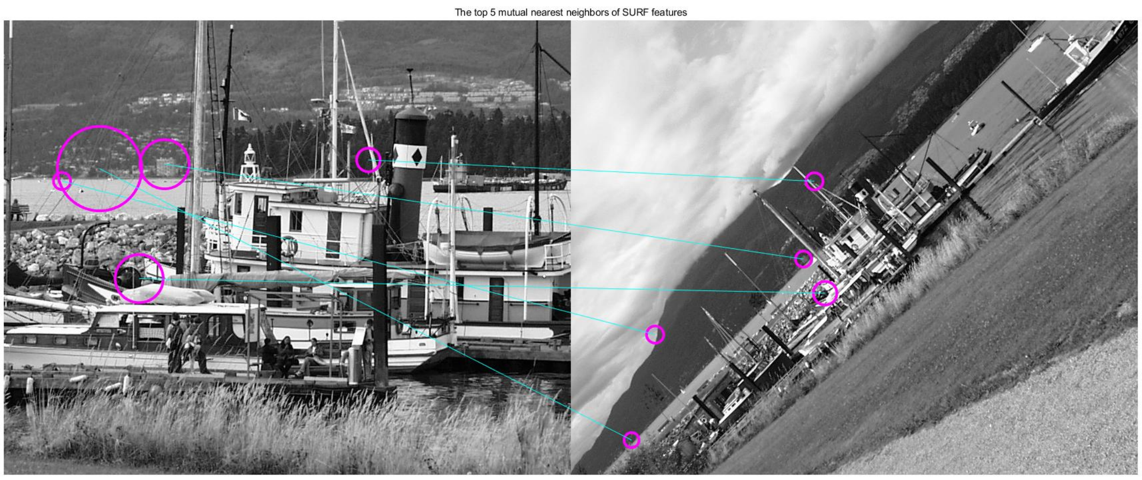

Figure 3: The best 5 matches using nearest neighbor distance.

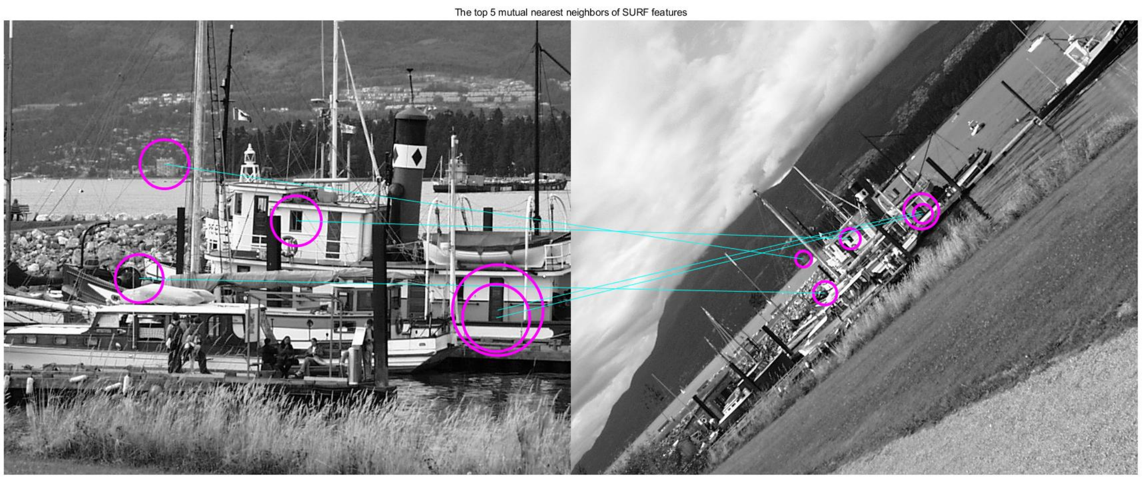

Figure 4: The best 5 matches using nearest neighbor distance ratio.

b) Harris corners and NCC based matching works great when image transformation is constrained with affine intensity change, image translation and image rotation, which is just the case for task 1. But in task 2, the features in second image are scaled and rotated compared with features in the first image, to which Harris corners is not robust anymore. In this case SURF is better than Harris corners. Because it's scale-invariant. Also NNDR is better than NCC because it will refuse the match if one of the two proposed patches have a second nearest neighbor with distance not so bigger than the distance with the nearest neighbor, which is a case for potential wrong match using NCC.
Harris corners would be better when we only wants a simple corner detector and need it to be fast and with less complexity. Because Harris Corners only gives the corner location but SURF also provides feature descriptors that can be used to compare keypoints.

Exercise 3. Scale-space blob detection.

The code for implementation can be found in **scaleSpaceBlobs.m**. Basic code can be found below.

```
for i=1:Nscales
    %fprintf('%d/%d\n',i,Nscales);
    sigmas(i)=k^(i-1)*sigma0;
    [g,gx,gy,gxx,gyy,gxy]=gaussian2(sigmas(i)^2);
    scaleNormalizedLaplacian=sigmas(i)^2*(gxx+gyy);
    % filter the image with the scale-normalized Laplacian of Gaussian
    % for each scale i and store the result to the
    %variable scalespace(:,:,i)
    %%-your-code-starts-here-%%
    %%-your-code-ends-here-%%
    filterd_img=imfilter(img,scaleNormalizedLaplacian,'same','symmetric');
    filterd_img=filterd_img.^2;
    scalespace(:,:,i)=filterd_img;
    %%-your-code-ends-here-%%
end
```

The best matches based on my own implementation and the pre-computed regions are shown separately in figure 5 and 6. The proposed circles in 2 images are shown in Figure 7 and 8

We can observe that the regions calculated by my own implementation are similar with the pre-computed regions.

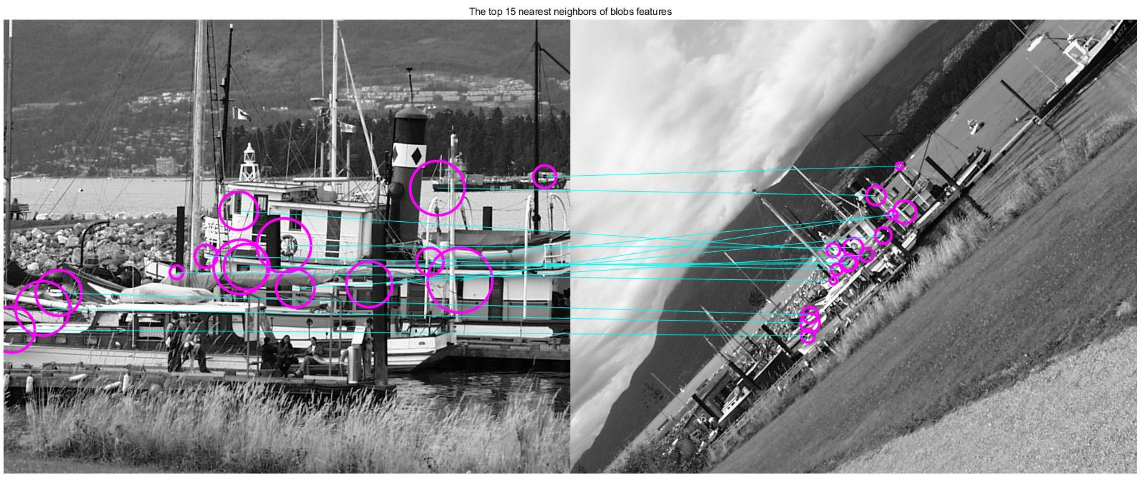

Figure 5: The best 20 matches based on pre-computed regions.

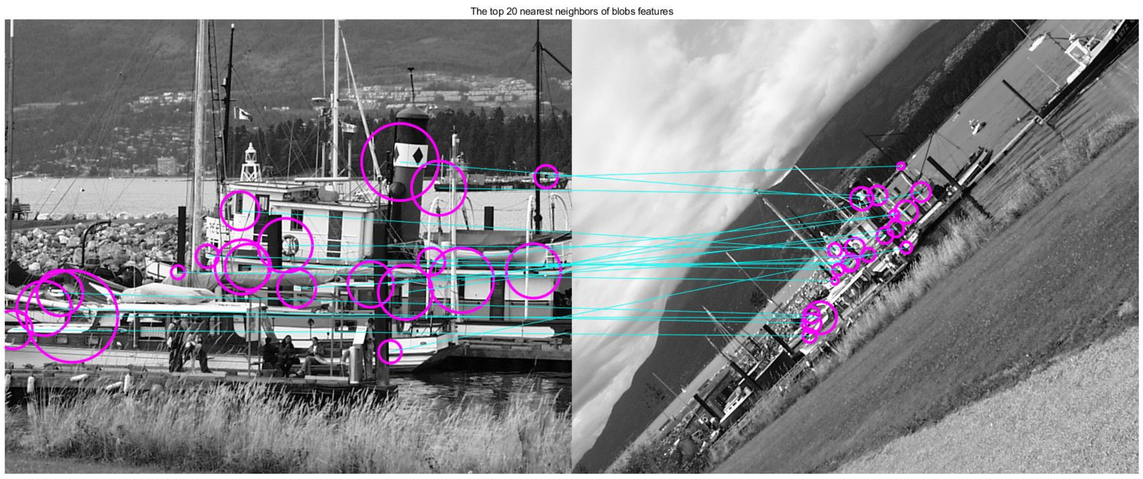

Figure 6: The best 20 matches based on self-implemented function**scaleSpaceBlobs.m**.

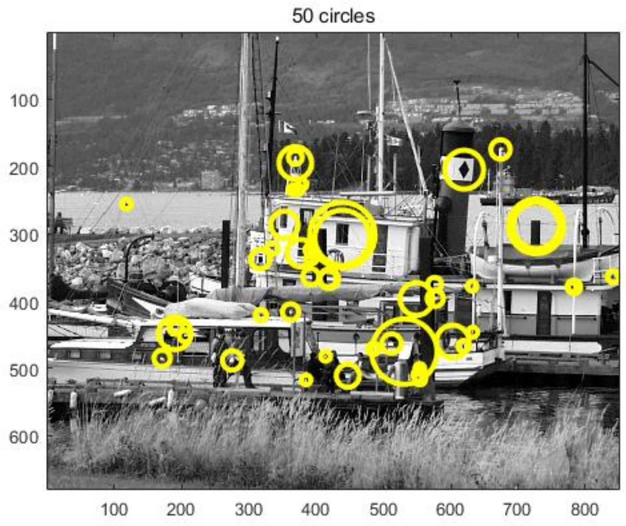

Figure 7: The proposed circles in the first image.

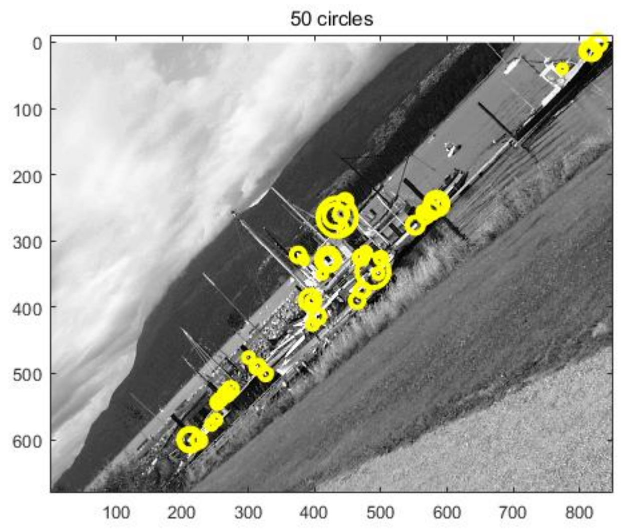

Figure 8: The proposed circles in the second image.