Computer Vision HW1

Anaelle Yasdi 203642129

Michael Novitsky 311773915

Part 1: Keypoint Detector

1.2

The Gaussian pyramid result is:

The equation for the Gaussian sigma parameter is:

$$\sigma_i = \sigma_0 \cdot k^{\text{levels}[i]}, \sigma_0 = 1, k = \sqrt{2}, \text{levels} = [-1\ 0\ 1\ 2\ 3\ 4]$$

As sigma increases, the gaussian kernel is wider which causes the image to be blurrier.

1.3

The Laplacian pyramid (the difference between two gaussians) result is:

We can see that different edge sizes are preserved for different sigma values.

1.4

In this part we calculated for each pixel in every DoG pyramid matrix the Principal curvature as following:

For every matrix corresponding to a Laplacian with a sigma value we calculated 3 matrices:

1. D_{xx} the second derivative in horizontal direction using Sobel filter
2. D_{xy} using Sobel filter
3. D_{yy} the second derivative in vertical direction using Sobel filter

The Hessian matrix was calculated for each pixel:

$$H(i,j) = \begin{bmatrix} D_{xx}(i,j) & D_{xy}(i,j) \\ D_{xy}(i,j) & D_{yy}(i,j) \end{bmatrix}$$

And the principal curvature was calculated as following:

$$R(i,j) = \frac{TR(H(i,j)^2}{Det(H(i,j)} = \frac{(\lambda_{min} + \lambda_{max})^2}{\lambda_{min} \cdot \lambda_{max}}$$

In total we calculated a principal curvature matrix for every DoG in the pyramid.

1.5

As we learned in class edges are not good key point to detect, and we needed to suppress the edges detected. We used the knowledge about the difference between edges and corners: edges has only one significant eigenvalue , and the other one is close to zero, whereas corners has both eigenvalues significantly larger then zero. This means that edges are expected to have larger principal curvature then corners.

In this part we created a method to detect the points of interest which satisfies 3 conditions:

1. They are local maxima's looking at their 8 neighbors in the same DoG scale and the 2 neighbors from a larger and smaller scale – a total of 10 neighbors. For simplicity we padded each DoG matrix with zeros to avoid corner cases.
   This condition allows us to detect a single point for each corner and not a "blob", like we saw in class.
2. They have a principal curvature value smaller than θ_r (meaning they are not edges)
3. They have a DoG magnitude larger then θ_c

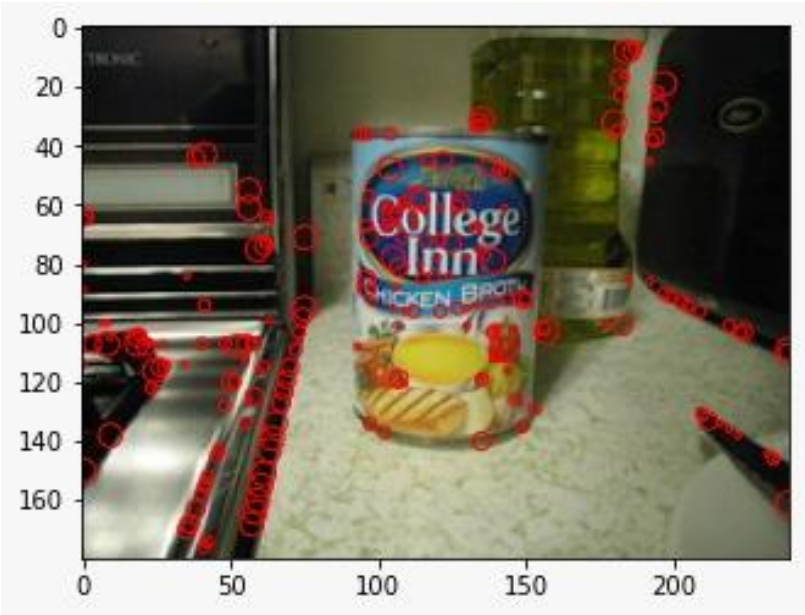
This method returns those points location (x,y) and scale in the DoG pyramid.

1.6

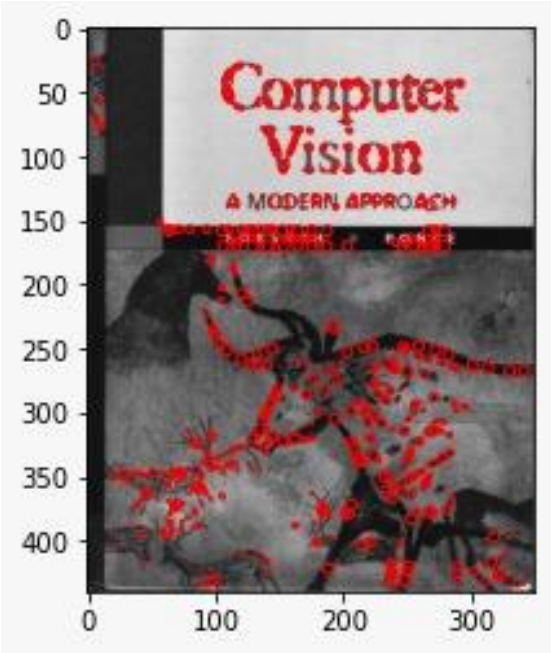
In this section we combined all the methods from previous sections and created full DoG detector, we then applied the detector on different images:

We can see the detection of interest points – some of them may correspond to edges but a scientific amount are valid interest points on 'blobs' inside the chicken broth can.

Looking at the same can with background we can notice similar points were detected (looking at the CHICKEN BROTH text).

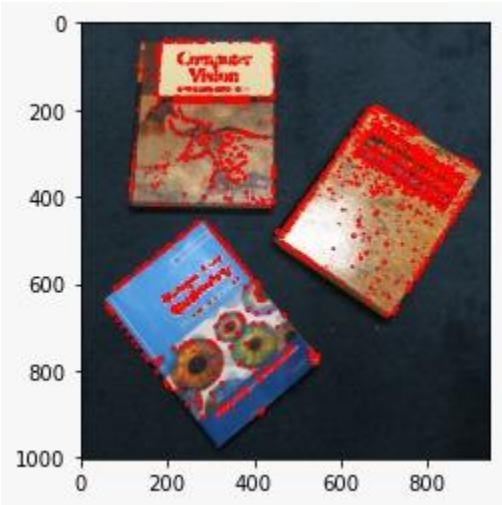


Looking at another example of the computer vision book



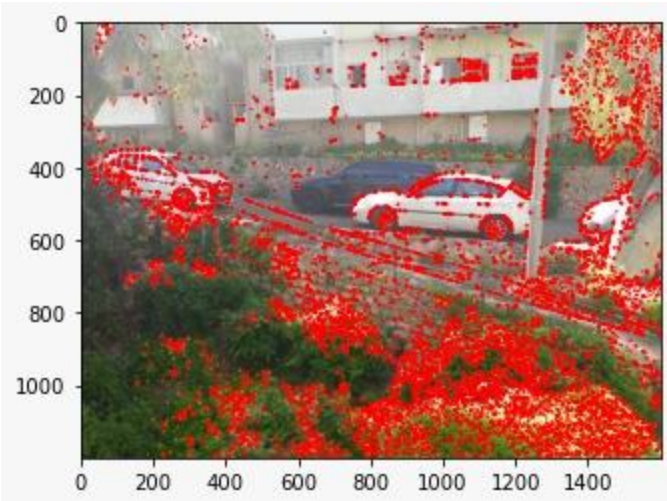
We can notice that most of detected points belong to letters and small areas in the cover image.

We tried the detector on the same book from a different angle:



We can still see that a lot of edges are detected, but also the same areas in the computer vision book are detected, even though the image was taken from different angle.

We also tried this detector on an outdoor image:



We can notice that must of the keypoints detected are concentrated in the bushes and trees. This is an outdoor image meaning that it contains a lot of different textures and objects to detect. Edge detection appears here in the fence line detection and the car edges.

Part 2: BRIEF Descriptor

2.1

In this section we created the indices vectors for pixels comparison in the BRIEF descriptor. Since the article presented that all the random sampling methods had shown good result, without a significant difference between them we chose to implement the uniformly distributed random method.

We saved the indices vectors for a 9x9 patch size and 256 $(\mathrm{x_n}, \mathrm{y_n})$ pairs.

2.2

In this part we created the descriptors matrix using this algorithm:

1. For each keypoint detected using DoG – go to the corresponding scale in the gaussian pyramid and take a 9x9 patch around the keypoint (x,y) location.
2. Calculate this patch descriptor vector as following:
   For all the (x_n, y_n) pairs from the indices vectors provided the descriptor in the n-th location value will be
   $$\rho(\mathrm{p}; \mathrm{x}, \mathrm{y}) = \begin{cases} 1 & , \mathrm{if\ p(x) < p(y)} \\ 0 & , \mathrm{otherwise} \end{cases}$$
3. Add this descriptor to the descriptors mat

The result will be a descriptors matrix with the descriptors as its rows and a keypoint location vector.

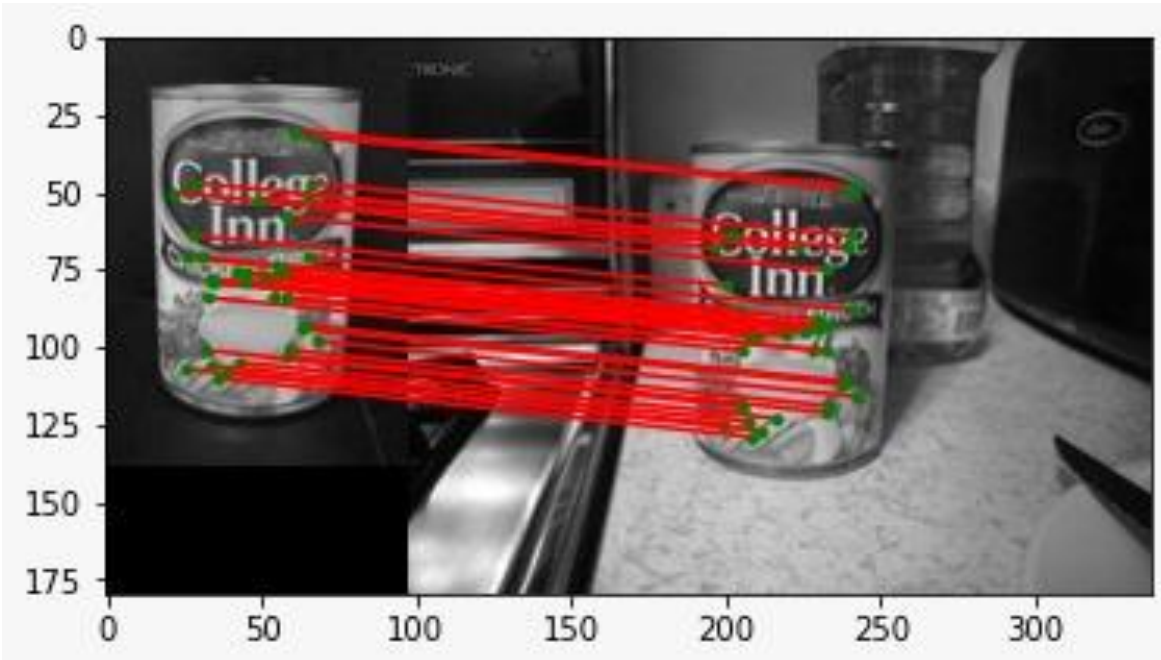
2.3

In this section we created the method to wrap all other methods needed for the BRIEF detector: this method calculates the image keypoints and their corresponding descriptors.

2.4

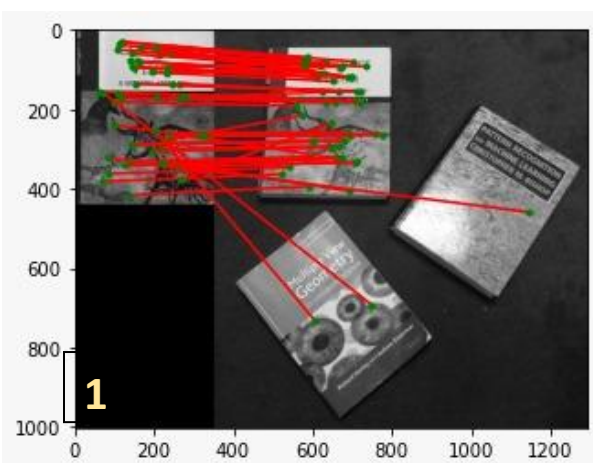
In the final section of this part we used the BRIEF descriptor to match two images of the same object using the provided function for descriptors matching.

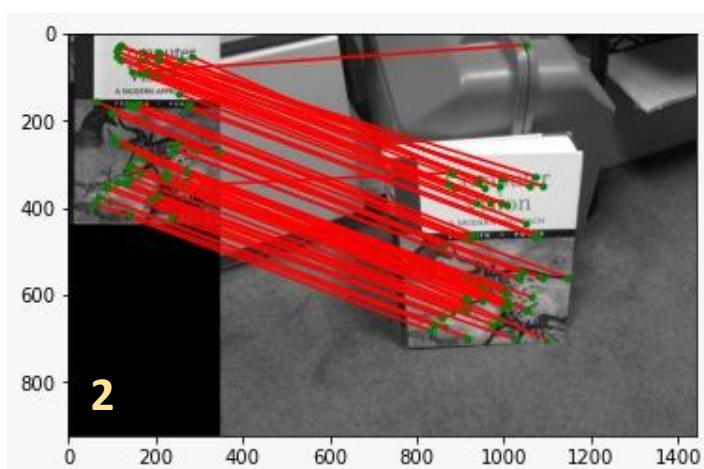
The first object we tested this algorithm was the chicken broth can:

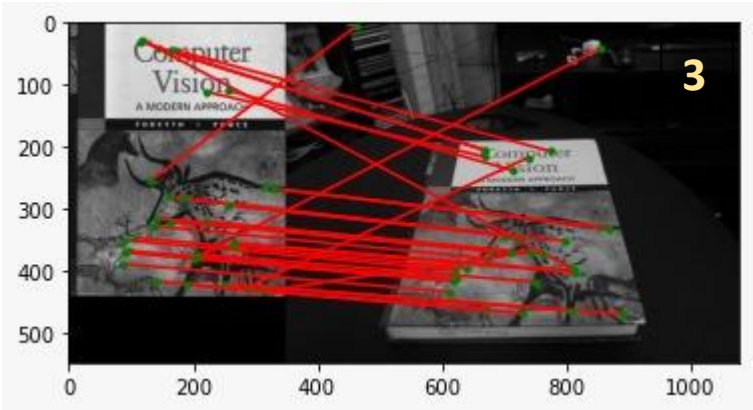


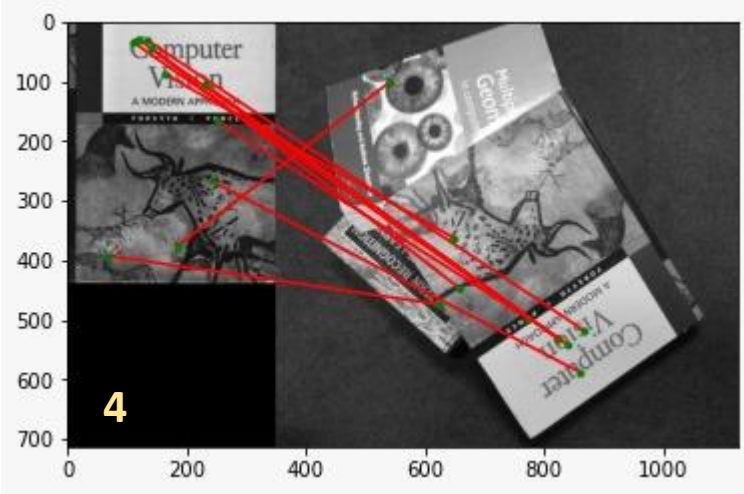
For images with approximately the same object scale, and orientation we get a good keypoint matching – we can see that the same areas around the text and image on the can are detected in both images.

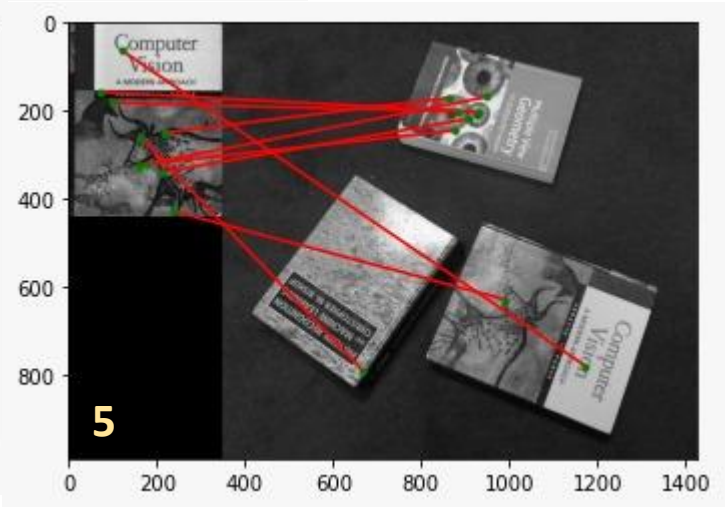
Another example is the computer vision book:





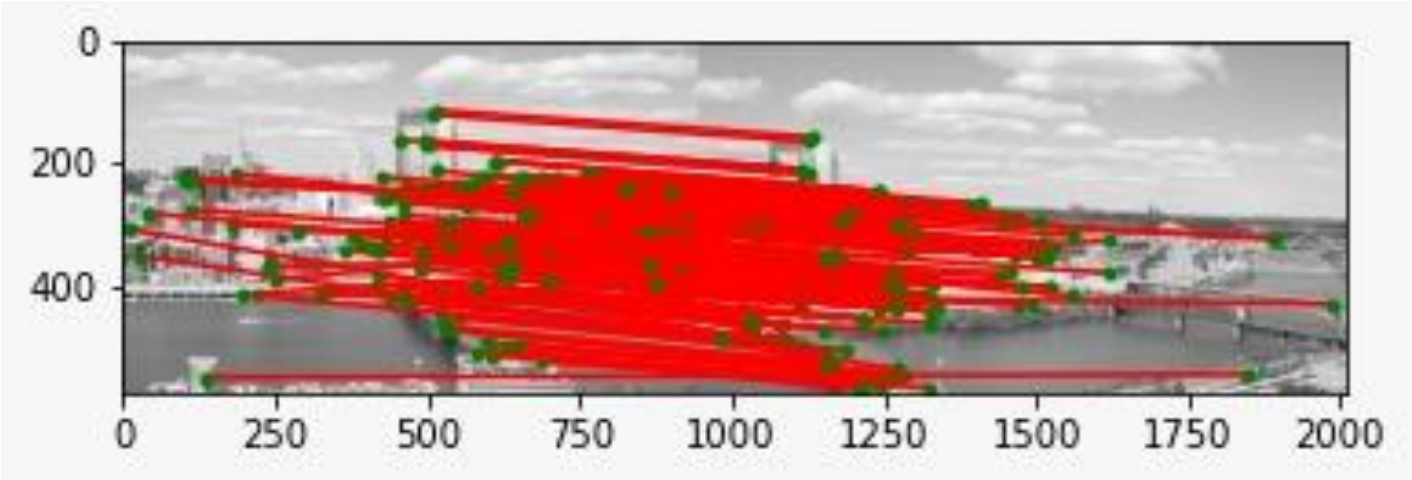






As we can see from the results this descriptor has good results with different scales and viewpoint if the same object (images (1),(2),(3)) , but it has poor results when the object is rotated (images (4),(5)). These results correspond to the expected results from the BRIEF descriptor properties we examined in class.

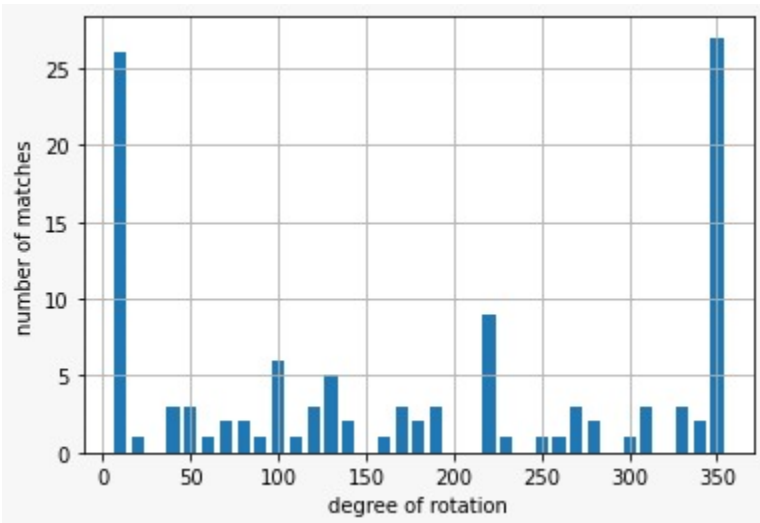
We also tested this descriptor with a landscape image:



As we discussed in previous sections, there are much more keypoints comparing to object images since this image contains different textures and objects. This fact makes it difficult to examine the quality of the matches, but if we focus on the top of the buildings where few objects and few keypoints exist, we can see a good match.

2.5 BONUS

In this section we were asked to rotate the chicken broth can in multiple angles and examine the number of correct matches.
The results are:



We can notice a significantly larger number of correct matches in the angles closer to 0 : 10 and 350, compared to the other angles. This result correlates to the conclusion from previous section – the BRIEF descriptor is not invariant to rotation as desired from a good descriptor.