Image Mining - Lab 1

Q1 EXPERIMENT the convolution code given as example in *Convolutions.py*. Note the difference between the direct calculation by scanning the 2d array and the calculation using function filter2d from OpenCV. Try to decrypt the OpenCV functions used for image reading and copying, and the MatPlotLib function used for image display.

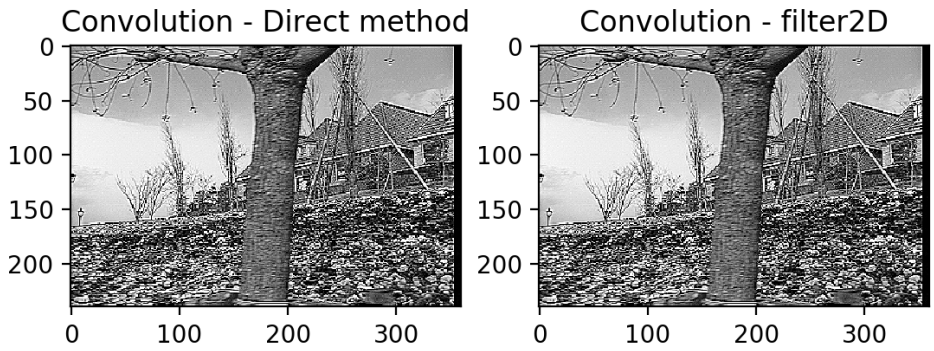

Figure 1 - result of Q1

Q2 EXPLAIN why the convolution kernel provided as example realizes a contrast enhancement with respect to the original image.

The new values of each pixel are calculated through the scalar product between the convolution kernel and the corresponding neighborhood of the pixel. First, the edge information in the image to have a higher contrast than the surrounding pixels. After the convolution, this contrast is further enhanced, so that the image looks sharp.

0	-1	0
-1	5	-1
0	-1	0

Figure 2 - convolution kernel

Q3 MODIFY the code to compute, with the two methods, the convolution that approximates the partial derivative $I_x = \frac{\partial I}{\partial x}$, then the gradient mod$\|\nabla I\| = \sqrt{I_x^2 + I_y^2}$ What precautions should be taken in order to get a correct display (i.e. a correct interpretation of the grayscales)?

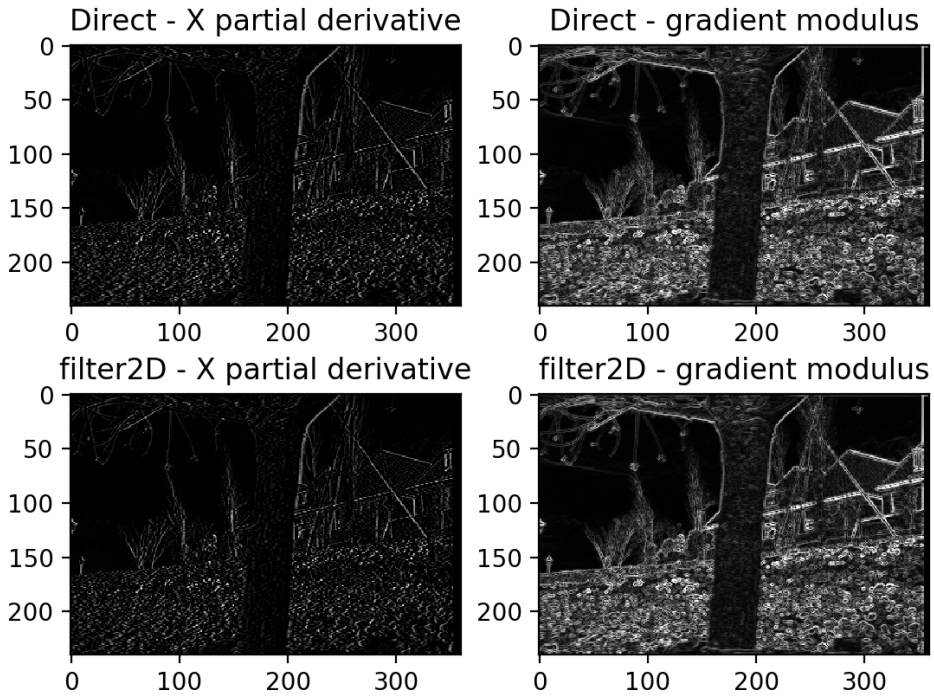

Figure 3 - x partial derivative and gradient modulus

Direct method: in every loop we do:
gx = img[y, x+1] - img[y, x-1] // x partial derivative
gy = img[y+1, x] - img[y-1, x] // y partial derivative
val = (gx**2+gy**2)**0.5 // gradient modulus
img2[y,x] = min(max(gx,0),255)
img3[y,x] = min(max(val,0),255)

Method filter2D:
kernelx = np.array([[-1, 0, 1]])
img4 = cv2.filter2D(img,-1,kernelx)
kernely = np.array([[-1], [0], [1]])
img5 = cv2.filter2D(img,-1,kernely)
img6 = np.sqrt(img4**2 + img5**2)

We need to limit the grayscale from 0 to 255 in order to get a correct display.

Q4 COMPLETE the code in the *Harris.py* script to compute the interest function of Harris (at one single scale), and the corresponding interest points. Explain

how the provided code, that uses morphologic dilation, allows to calculate the local maxima of the interest function Theta.

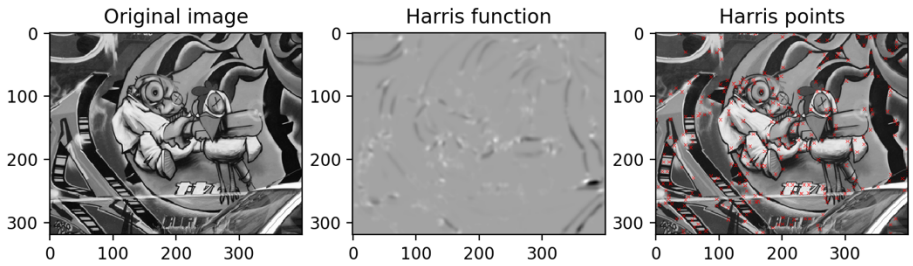

Figure 4 - Harris scale = 3, α = 0.06

```
scale = 3
k = 0.06
gx, gy = np.zeros(img.shape), np.zeros(img.shape)
filters.gaussian_filter(img, (scale, scale), (0, 1), gx)
filters.gaussian_filter(img, (scale, scale), (1, 0), gy)

xx = filters.gaussian_filter(gx*gx, scale)
xy = filters.gaussian_filter(gx*gy, scale)
yy = filters.gaussian_filter(gy*gy, scale)

det = xx * yy - xy ** 2
trace = xx + yy
Theta = det - k * trace ** 2
```

In this case, d_maxloc = 3, so with the function “cv2.dilate”, the local maximum point is the same as the original while other non-local maximum points are replaced by the maximum points in the 3 * 3 neighborhood.

Q5 COMMENT the results obtained with your Harris detector and the effect of the used parameters, in particular the size of the summing window and the value of α. How is it possible to extend this computation on several scales?

When α is constant, the larger the scale is, the more blurred the image is. It will find more corner points. When scale is constant, the larger α is, the whiter the image is. When α is too large, we can’t find the corner points (shown in Figure 7). So scale = 3 and α = 0.06 are appropriate.

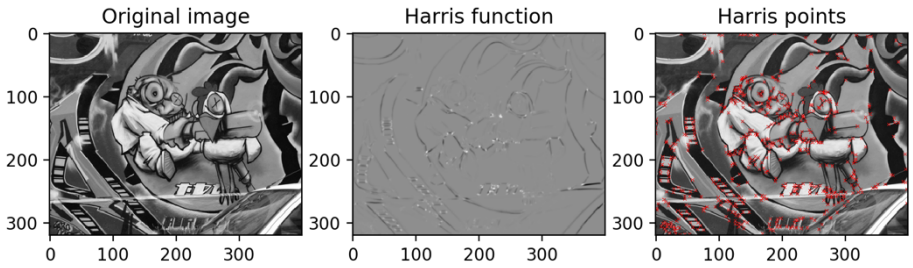

Figure 5 - scale = 1, $\alpha = 0.06$

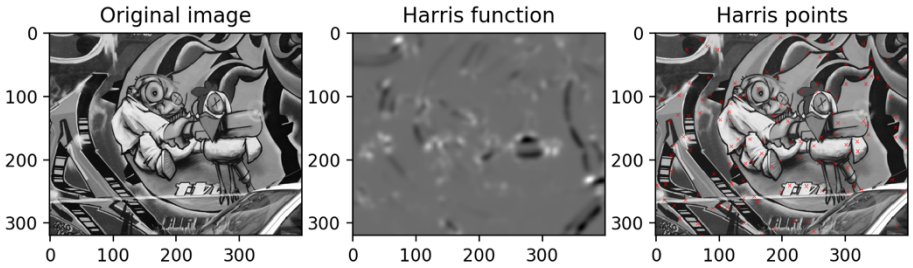

Figure 6 - scale = 5, $\alpha = 0.06$

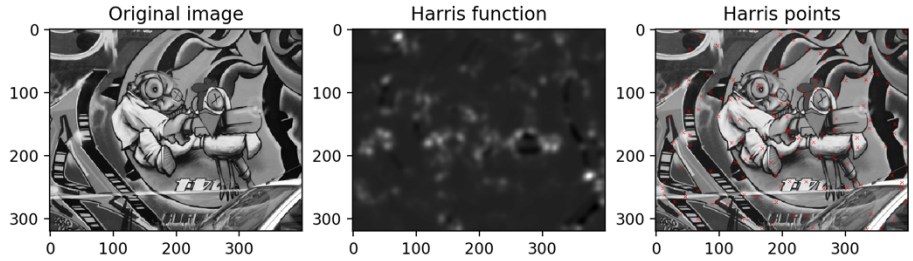

Figure 7 - scale = 3, $\alpha = 0.02$

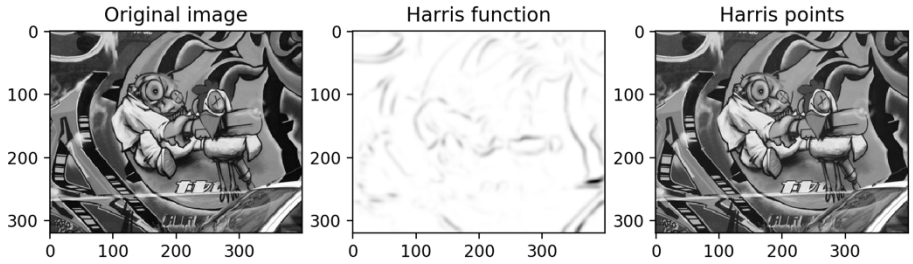

Figure 8 - scale = 3, $\alpha = 0.6$

Q6 EXPERIMENT and compare the two detectors ORB and KAZE by running the script *Features_Detect.py*. Find the principle of each detector. Explain the main parameters intrinsic to each detector and their effect on the detection. How is it possible to visually evaluate the repeatability of each detector applied on a pair of images?

ORB (Oriented FAST and Rotated BRIEF), is a combination and improvement of FAST (Features from Accelerated Segment Test) feature detection and BRIEF feature descriptor. It uses the method of FAST feature detection to detect feature points, and

then uses the Harris corner score to select top quality points with the largest response value.

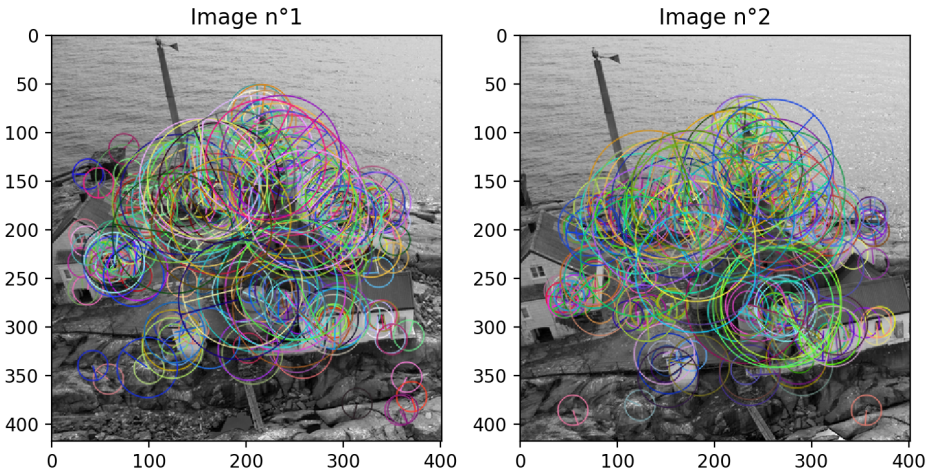

Figure 9 - ORB

The main parameters of ORB:

1. nfeatures - Maximum number of features (keypoints) to find.
2. scaleFactor - Pyramid extraction rate must be greater than 1. The ORB uses image pyramids to find features, so it must provide a scale factor between each layer in the pyramid and the number of levels the pyramid has. scaleFactor = 2 represents a classic pyramid where each next level of pixels is 4 times lower than the previous level. Large scale factors will reduce the number of features found.
3. nlevels - The number of pyramid levels.

KAZE detector is based on scale normalized determinant of Hessian Matrix which is computed at multiple scale levels. The maxima of detector response are picked up as feature-points using a moving window. KAZE uses AOS (Additive Operator Splitting) algorithm and variable conduction to establish a non-linear scale space and proposes a new concept of evolution time.

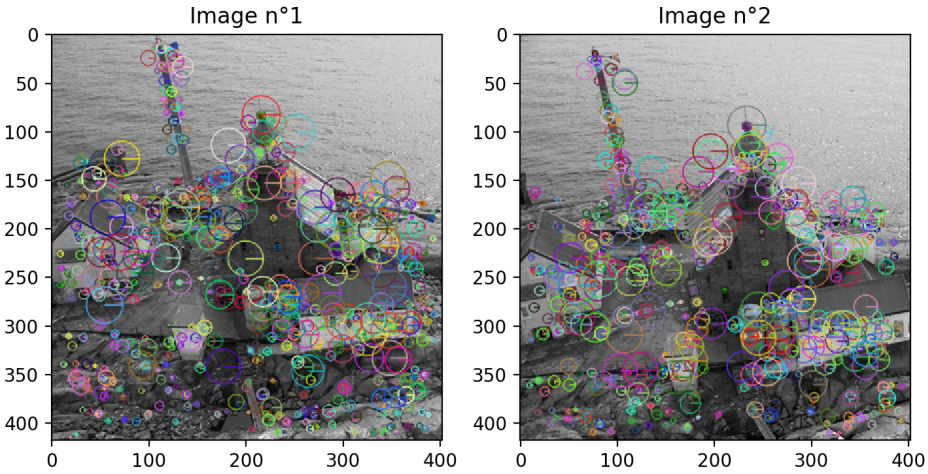

Figure 10 - KAZE

The main parameters of KAZE:

1. upright - Set to enable use of upright descriptors (non-rotation invariant)
2. threshold - Detector response threshold to accept point
3. nOctaves - Maximum octave evolution of the image
4. nOctaveLayers - Default number of sublevels per scale level

Repeatability of a feature-detector is the percentage of detected features that survive photometric or geometric transformations in an image. It only depends on performance of the feature-detection part of feature-detection-description algorithm. It is calculated on the basis of the overlapping (intersecting) region in the subject image pair. A feature-detector with higher repeatability for a particular transformation is considered robust than others, particularly for that type of transformation.

Q7 EXPLAIN the principle of the descriptors attached to points ORB and those attached to points KAZE. What properties of detectors and/or descriptors (distinguish the two aspects in your answer), allow to make the matching scale and rotation invariant?

ORB: BRIEF (Binary Robust Independent Elementary Features), n point pairs are randomly selected in the neighborhood of the feature points, and then an n-dimensional binary descriptor is formed. Because BRIEF description method has no rotation invariance, we use a modified version of BRIEF descriptor, “Steer BRIEF”, which makes BRIEF has rotation invariance. The speed of the ORB algorithm is by far the fastest, it can be applied real-time, and has good rotation invariance and noise resistance. However, the ORB algorithm does not solve the problem of scale

invariance, because FAST itself does not have scale invariance.

KAZE: M-SURF descriptors. Feature description introduces the property of rotation invariance by finding dominant orientation in a circular neighborhood around each detected feature. KAZE features are invariant to rotation, scale, limited affine and have more distinctiveness at varying scales with the cost of moderate increase in computational time. The running speed of the KAZE algorithm is fast, and the extracted feature points have good stability. Even if the image undergoes certain deformation, the effect on the feature points is not large.

Q8 EXPLAIN and compare qualitatively the effects of the three different point matching strategies performed in the three scripts *Features_Match_CrossCheck.py*, *Features_Match_RatioTest.py* and *Features_Match_FLANN.py*. Explain why the used distances are different for the two descriptors. Explain why the strategy FLANN (and, in a lesser extent the strategy RatioTest) does not work well with ORB points.

When we use ORB, in the picture of crossCheck, we can see many mistakes in matching. And in the picture of ratioTest, it missed many points. The point matching strategies performed not very well.

As for the KAZE, the three point matching strategies showed similar performances.

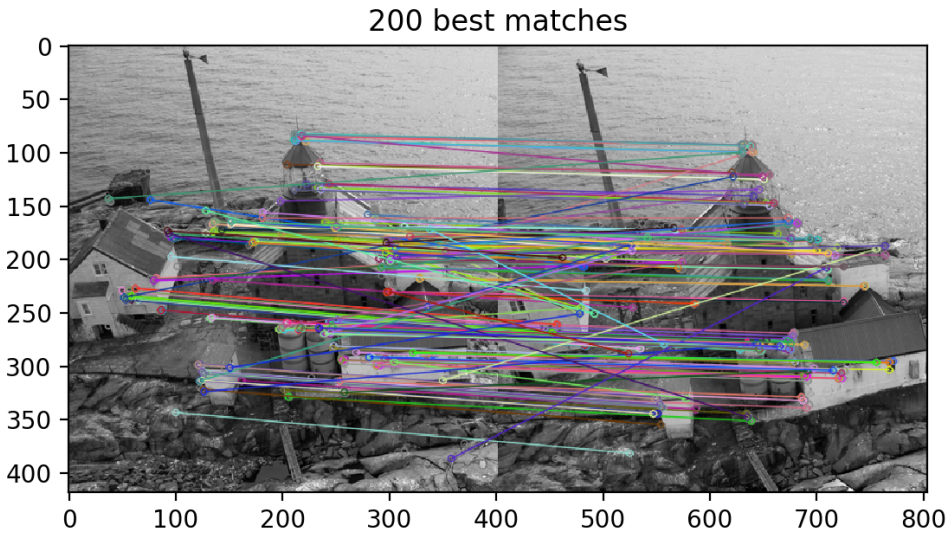

Figure 11 - CrossCheck ORB

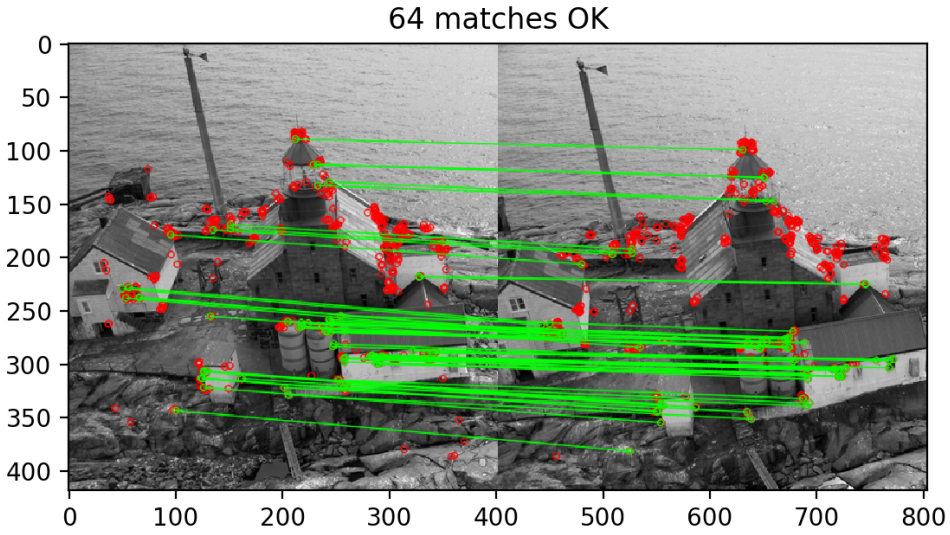

Figure 12 - RatioTest ORB

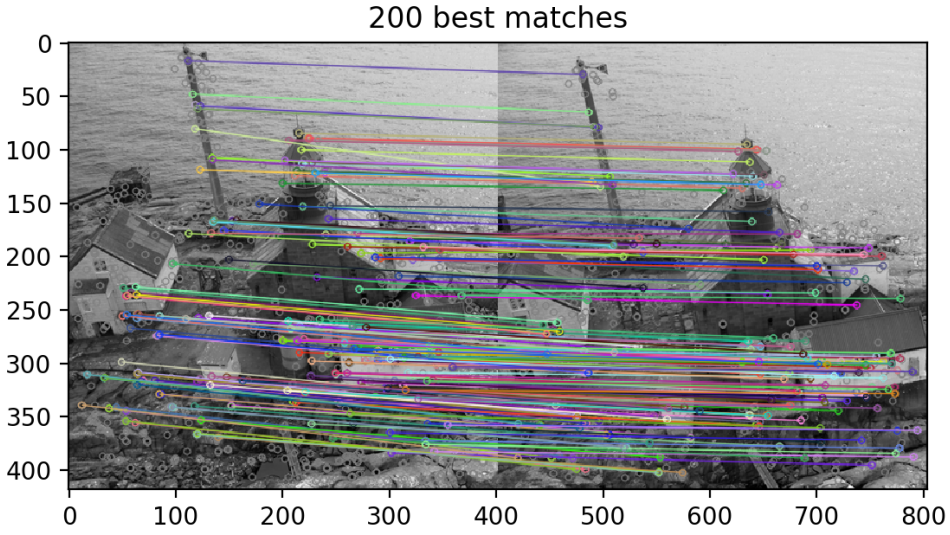

Figure 13 - CrossCheck KAZE

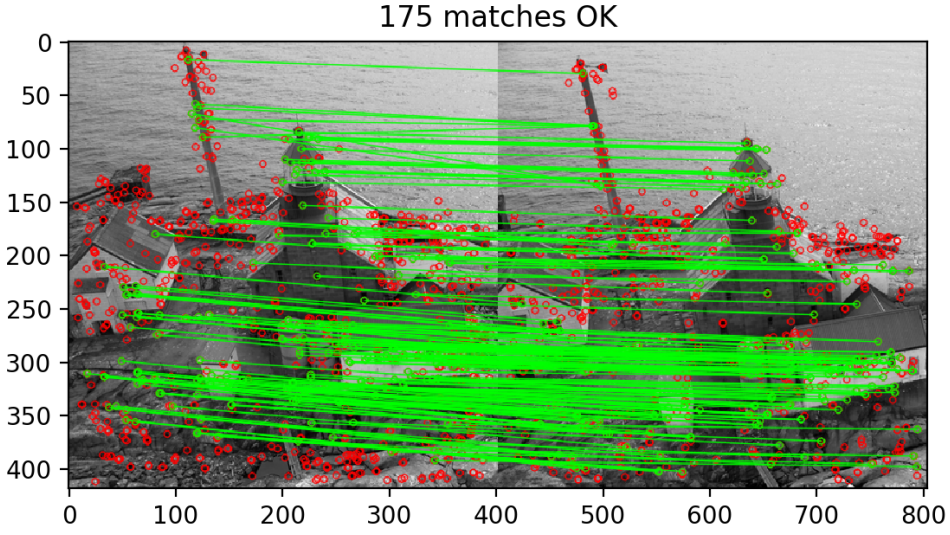

Figure 14 - RatioTest KAZE

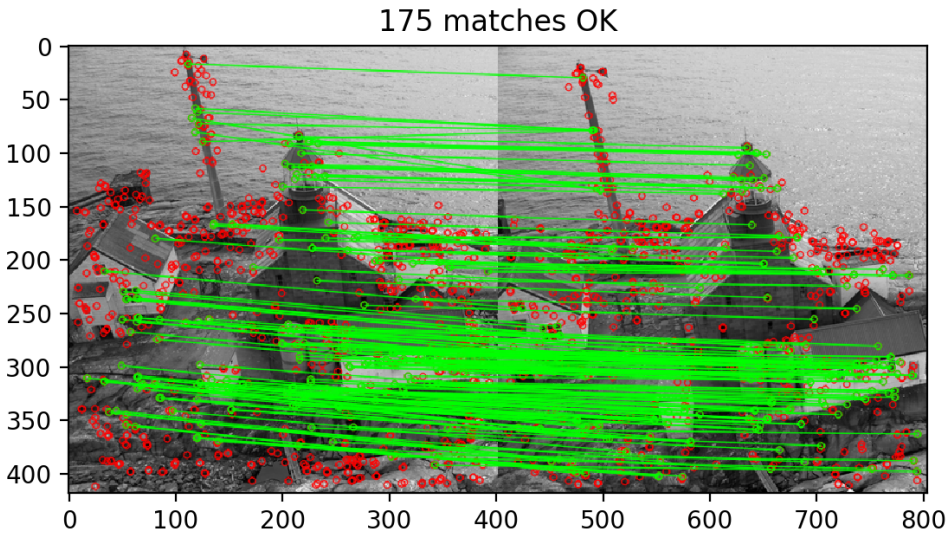

Figure 15 - FLANN KAZE

Since BRIEF descriptor uses binary ("0" and "1") encoding, only the Hamming distance of the two feature point descriptors need to be calculated during feature matching. And KAZE use the most common distance, Euclidean distance.

ORB uses binary encoding, but FLANN calculates distances based on float, so there exists a data type mismatch error.

Q9 PROPOSE a strategy to evaluate quantitatively the matching by using an image with a known geometric transformation (OpenCV functions like cv2.warpAffine can be used).

First, we do geometric transformations such as affine and illumination transformation on the original image to obtain multiple test images. Then find the repetition rate of the original image a and the test image b for the set R, which indicates the ratio of the number of correctly matched points to the number of feature points extracted.