Feature and Image matching

Objective

To understand the concept of Feature detectioin and matching

Main ideas

I. Feature Detection

1. Feature:

a. Definition: A feature is a piece of information, it may be a structure of images such as edges or objects. It is used to solve task in computer applications. The feature can be classified into two categories:

- Keypoint features: in a specific locations of an image (peeks, corners, etc.)
- Edges: can be matched based on orientation and appearance.

We can detect these feature via **Interest Point** - the point which is expressive in texture, at which the direction of the boundary of the object changes abruptly.

b. Why: Features are used for:

- Image alignment. Eg., panorama
- Object recognition
- 3D reconstruction
- Motion tracking
- Robot navigation

c. Methods:

In this document, we shall focus on **Scale Invariant Feature Transfor (SIFT)** and **The Harris operator**. Introduced by Chris Harris and Mike Stephens in 1988, Harris Corner Detector is a corner detection operator mostly used to extract corners and infer features of images. To extract corners, we consider a small window around each pixel to identify whether it is unique or not by measuring the amount of changes in pixel values or shifting window by a small amount in a given direction.

To do this, we define a function called the **sum** of all the sum squared differences (SSD) *E(u, v)*:

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u,y+v) - I(x,y)]^2$$

The equation

- u, v: x, y coordinates of every pixel in 3x3 window.
- I: the intensity value of the pixel.

Minimizing this function using Taylor Expansion, we get:

$$E(u,v) \approx [u \quad v]\left(\sum \begin{bmatrix} {I_x}^2 & I_xI_y \\ I_xI_y & {I_y}^2 \end{bmatrix}\right)\begin{bmatrix} u \\ v \end{bmatrix}$$

The matrix can be renamed as M, we now get a better function:

$$E(u,v) \approx [u \quad v]\, M \begin{bmatrix} u \\ v \end{bmatrix}$$

A number R is calculated for each window:

$$R = \det M - \ k(trace\, M)^2$$

$$\det M = \lambda_1 \lambda_2$$

$$trace\, M = \lambda_1 + \lambda_2$$

where λ1 and λ2 are the eigenvalues of M, this value decides whether a region is a corner, edge or flat:

- When |R| is small, the region is flat.
- When R < 0, the region is an edge.
- When R is large, the region is corner.

SIFT is a feature detection algorithm published by David Lowe in 1999 to detect and describe local features in images. This method has more advantages than the Harris operator since it is not scale invariant, those advantages are:

- Locality: robust to occlusion and clutter.
- Distinctiveness: individual features can be matched to a large database of objects.
- Quantity: many features can be generated even small objects.
- Efficiency: close to real-time performance.
- Extensibility: can easily be extended.

There are mainly four steps involved:

- Scale-space peak selection: Potential location for finding features.
- Keypoint Localization.
- Orientation Assignment: Assign to keypoints.
- Keypoint descriptor: Describing the keypoints as a high dimensional vector.
- Keypoint Matching.

II. Image Matching

Features matching or image matching is the task establishing correspondences between two images of the same scene. There are two main algorithms to do this which are: **Brute-force Matcher** or **FLANN (Fast Library for Approximate Nearest Neighbors) Matcher**. The algorithm for feature detection and then matching are:

- Find a set of distinctive keypoints.
- Define a region around each keypoint.
- Extract and normalize the region content.
- Compute a local descriptor from the normalized region.
- Match local descriptors.

We shall use Brute-force matcher as an example. The method is fairly simple as it takes the descriptor of one feature in first image and matched with all other features in the second image. The closest matched will be returned. It can also implement SIFT and ORB to match.

Examples in OpenCV

Feature Detection

Code

```
import numpy as np
import cv2 as cv
import sys

forme1 = cv.imread(cv.samples.findFile("Resource/forme1.png"))
lena = cv.imread(cv.samples.findFile("Resource/lena.png"))

gray= cv.cvtColor(forme1,cv.COLOR_BGR2GRAY)
gray2 = cv.cvtColor(lena, cv.COLOR_BGR2GRAY)
gray = np.float32(gray)

dst = cv.cornerHarris(gray,2,3,0.04)
dst = cv.dilate(dst,None)

forme1[dst>0.01*dst.max()]=[0,0,255]

sift = cv.SIFT_create()
kp = sift.detect(lena, None)

img = cv.drawKeypoints(gray2, kp, lena)

cv.imshow('Forme 1', forme1)
cv.imshow("Lena", img)

key = cv.waitKey(0)
if key == ord("q"):
    sys.exit(0)
```

Result

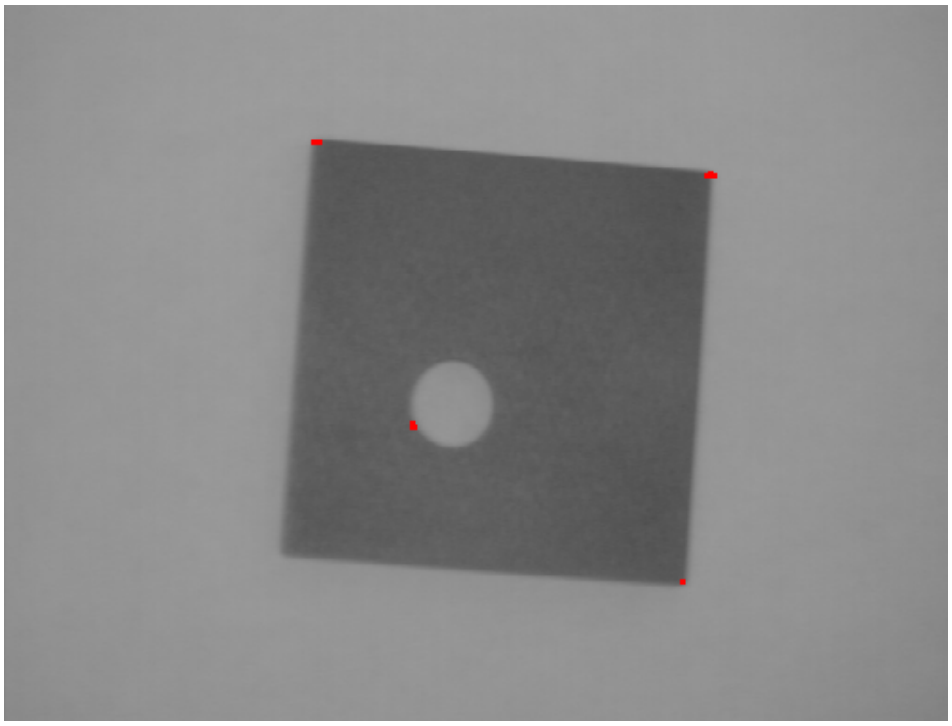
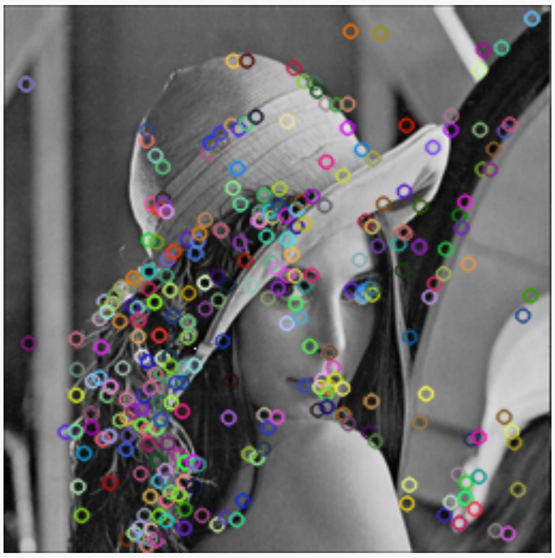
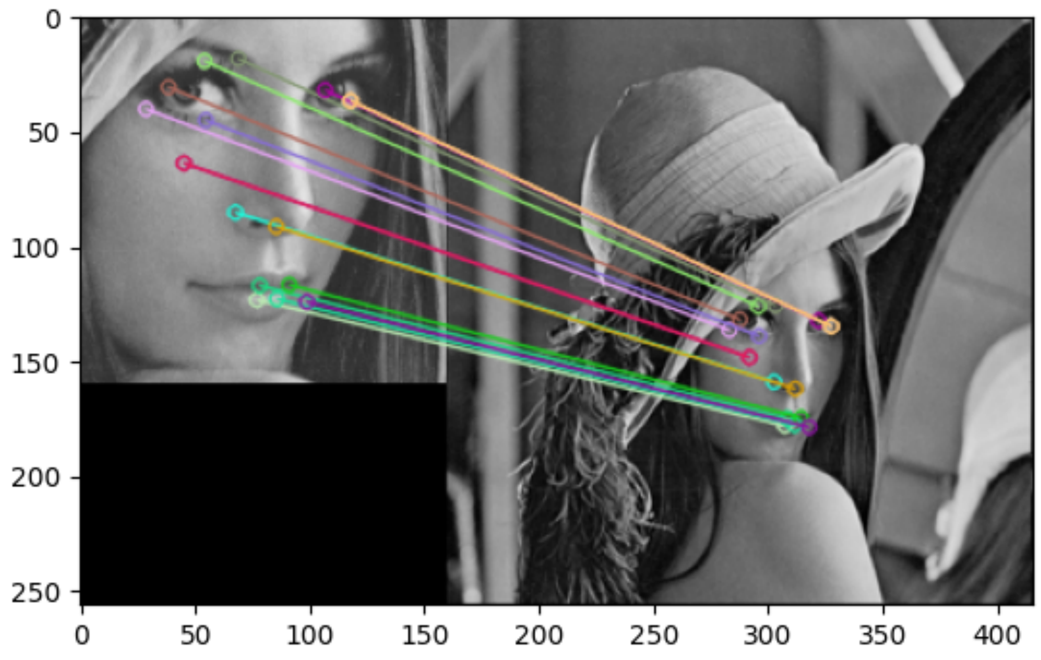
Method	Outcome
Harris operator	
SIFT	

Image Matching

Code

```
import sys
import cv2 as cv
from matplotlib import pyplot as plt

img1 = cv.imread(cv.samples.findFile("Resource/lena_face.png.png"), cv
.IMREAD_GRAYSCALE)
img2 = cv.imread(cv.samples.findFile("Resource/lena.png"), cv.IMREAD_G
RAYSCALE)

sift = cv.SIFT_create()

kp1, des1 = sift.detectAndCompute(img1,None)
kp2, des2 = sift.detectAndCompute(img2,None)

bf = cv.BFMatcher()
matches = bf.knnMatch(des1,des2,k=2)

good = []
for m,n in matches:
    if m.distance < 0.75*n.distance:
        good.append([m])

img3 = cv.drawMatchesKnn(img1,kp1,img2,kp2,good,None,flags=cv.DrawMatc
hesFlags_NOT_DRAW_SINGLE_POINTS)
plt.imshow(img3),plt.show()

key = cv.waitKey(0)
if key == ord("q"):
    sys.exit(0)
```

Result



Conclusion

Feature and Image matching can be very useful in daily life as it is an important task in many computer apllications namely object detection, image retrieval and so on.