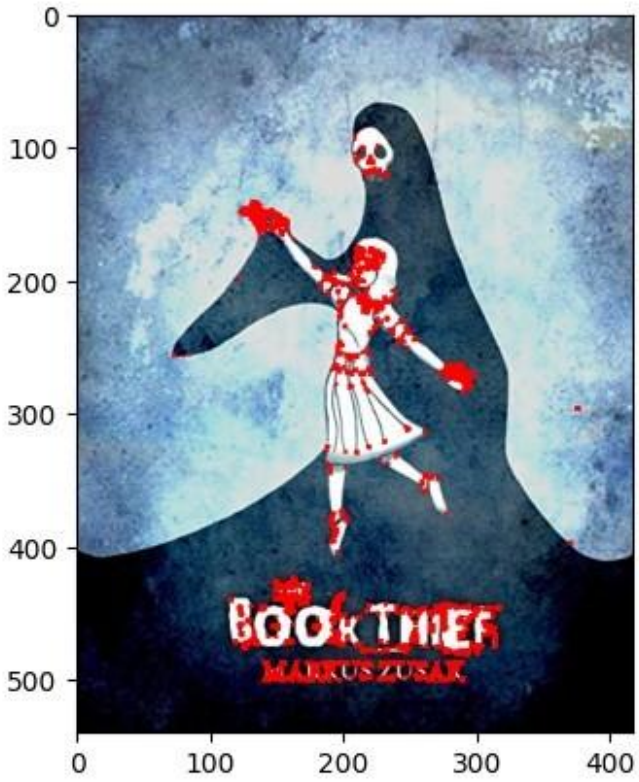
# WEEK-4

**Harris Corner Edge Detection:**

Harris Corner Detector is the improvement of Moravec's corner detector. Compared to the previous one, Harris' corner detector takes the differential of the corner score into account with reference to direction directly, instead of using shifting patches for every 45-degree angles, and has been proved to be more accurate in distinguishing between edges and corners.

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
# %matplotlib inline
filename = 'the_book.jpg'
img = cv2.imread(filename)
gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
gray = np.float32(gray) dst =
cv2.cornerHarris(gray,2,3,0.04) dst =
cv2.dilate(dst,None)
img[dst>0.01*dst.max()]=[255,0,0]
plt.imshow(img)
```




**SIFT(Scale Invariance Feature Transform):**

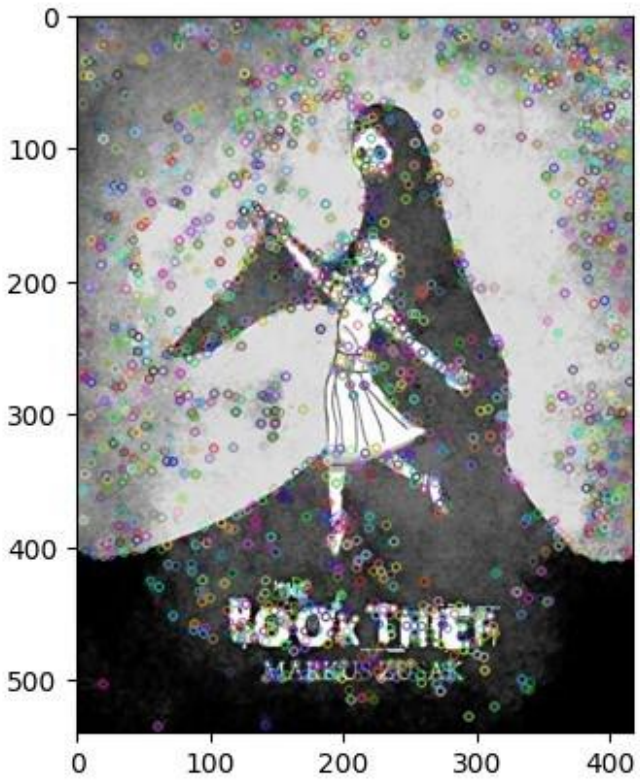
SIFT is invariance to image scale and rotation.

This algorithm is patented, so this algorithm is included in the Non-free module in OpenCV.

**SIFT for a Single image:**

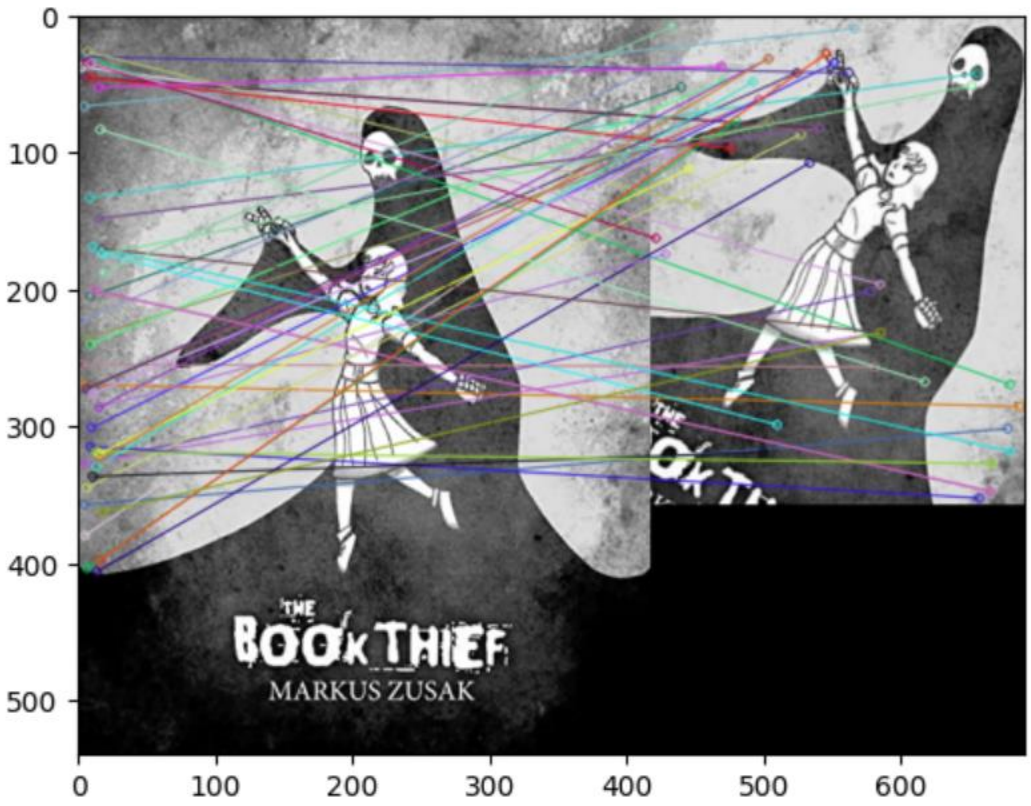
```
img = cv2.imread("the_book.jpg", cv2.IMREAD_GRAYSCALE)
sift = cv2.xfeatures2d.SIFT_create()
keypoints_sift, descriptors = sift.detectAndCompute(img, None)
img1 = cv2.drawKeypoints(img, keypoints_sift, None)
plt.imshow(img1)
```

```
<matplotlib.image.AxesImage at 0x1b247643160>
```




**SIFT for comparing Two images:**

```
img1 = cv2.imread("the_book.jpg", cv2.IMREAD_GRAYSCALE)
img2 = cv2.imread("the_book1.jpg", cv2.IMREAD_GRAYSCALE)
sift = cv2.xfeatures2d.SIFT_create()
keypoints_sift1, descriptors1 = sift.detectAndCompute(img1, None)
keypoints_sift2, descriptors2 = sift.detectAndCompute(img2, None)
bf=cv2.BFMatcher()
matches=bf.match(descriptors1,descriptors2)
img3=cv2.drawMatches(img1,keypoints_sift1,img2,keypoints_sift2,matches
[:50],None,flags=2)
plt.imshow(img3)
```

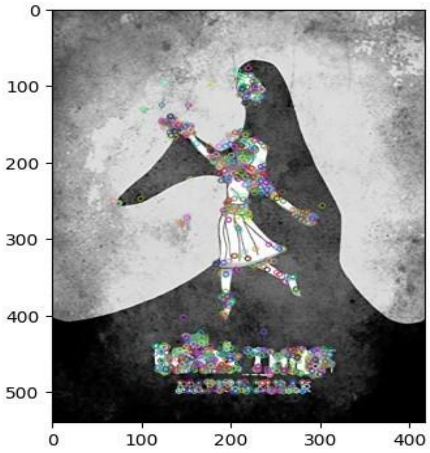



**ORB(Oriented FAST and rotated BRIEF)**:

ORB performs as well as SIFT on the task of feature detection (and is better than SURF) while being almost two orders of magnitude faster. ORB builds on the well-known FAST keypoint detector and the BRIEF descriptor.

**ORB for Single Image:**

```
img = cv2.imread("the_book.jpg", cv2.IMREAD_GRAYSCALE)
orb = cv2.ORB_create(nfeatures=1500)
keypoints_orb, descriptors = orb.detectAndCompute(img, None)
img1 = cv2.drawKeypoints(img, keypoints_orb, None)
plt.imshow(img1)
```



ORB for comparing of Two Images:

```
img1 = cv2.imread("the_book.jpg", cv2.IMREAD_GRAYSCALE)
img2 = cv2.imread("the_book1.jpg", cv2.IMREAD_GRAYSCALE)
orb = cv2.ORB_create(nfeatures=1500)
keypoints_orb1, descriptors1 = orb.detectAndCompute(img1, None)
keypoints_orb2, descriptors2 = orb.detectAndCompute(img2, None)
bf=cv2.BFMatcher(cv2.NORM_HAMMING,crossCheck=True)
matches=bf.match(descriptors1,descriptors2) matches = sorted(matches,
key = lambda x:x.distance)
img3=cv2.drawMatches(img1,keypoints_orb1,img2,keypoints_orb2,matches[:
50],None,flags=2)
plt.imshow(img3)
<matplotlib.image.AxesImage at
0x1b247eb19f0>
```