Date : 03/09/2025

FCV WEEK5 SUBMISSION

Name: Davasam Karthikeya   Section: AIMLB   Reg No: 230962326

1. Implement basic feature extraction algorithms given below:

i **Harris corner detection**

ii FAST corner detection.

Apply it to different images and visualize the detected keypoints.

Code:-

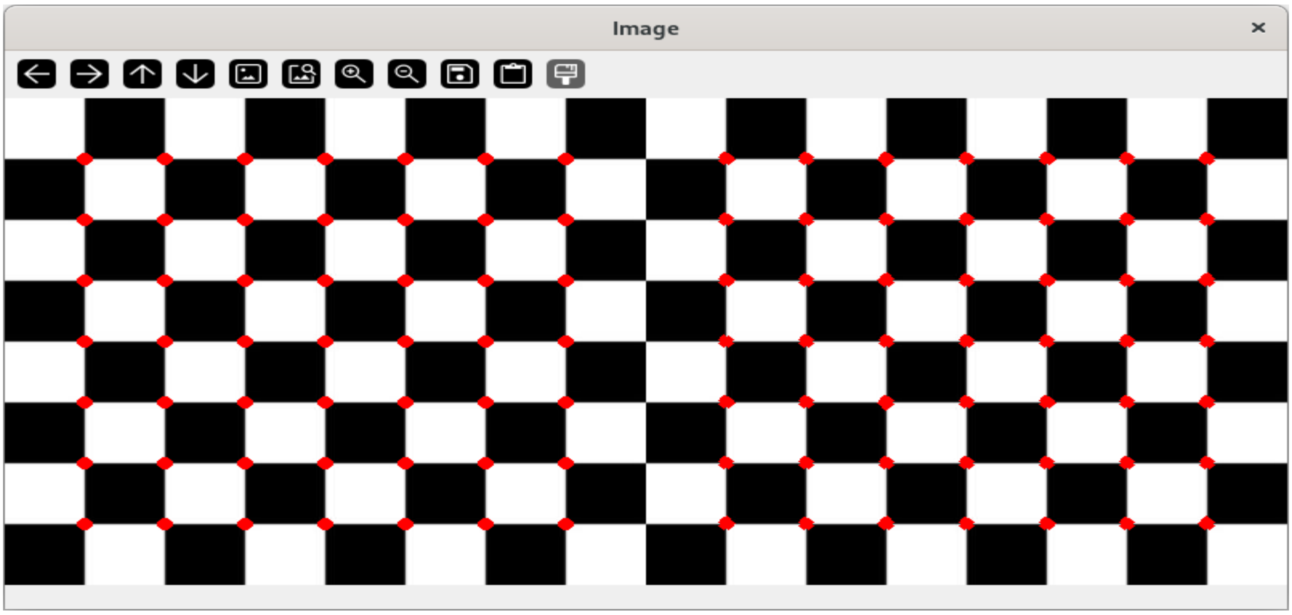
```
import cv2 as cv
import numpy as np

img = cv.imread('Week5/Chess_Board.jpg')

def Harris_Corners(img:cv.Mat, ksize=3, threshold=0.01):
    img_gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
    gx, gy = cv.Sobel(img_gray, cv.CV_64F, 1, 0), cv.Sobel(img_gray, cv.CV_64F,
0, 1)

    Ixx = gx**2
    Iyy = gy**2
    Ixy = gx*gy

    height, width, _ = img.shape
    n = (ksize-1)//2

    Neighbours = [(1, 0), (-1, 0), (0, 1), (0, -1), (1, 1), (-1, -1), (1, -1),
(-1, 1)]

    R = np.zeros((height, width), dtype=np.float64)

    for i in range(n, height-n):
        for j in range(n, width-n):
            ixx, iyy, ixy = 0, 0, 0
            for k in range(1, n+1):
                for dx, dy in Neighbours:
                    ixx += Ixx[i + dx*k][j + dy*k]
                    iyy += Iyy[i + dx*k][j + dy*k]
                    ixy += Ixy[i + dx*k][j + dy*k]

            R[i][j] = (ixx*iyy - (ixy**2)) - (ixx + iyy)

    R = cv.normalize(R, None, 0, 1, cv.NORM_MINMAX)
    out = img.copy()

    for i in range(height):
        for j in range(width):
            if R[i, j] > threshold:
                cv.circle(out, (j, i), 2, (0, 0, 255), -1)

    return out

def FAST_Corners(img:cv.Mat, n=12, treshold=10):
```

Date : 03/09/2025

```
    img_gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
    height, width, _ = img.shape

    miniCircle = [(-3, 0), (3, 0), (0, -3), (0, 3)]
    bresenhamCircle = [(0, 3), (1, 3), (2, 2), (3, 1), (3, 0), (3, -1), (2, -2),
(1, -3), (0, -3), (-1, -3), (-2, -2), (-3, -1), (-3, 0), (-3, 1), (-2, 2), (-1,
3)]

    def isActivePixel(I, curI, treshold):
        return I > curI + treshold or I < curI - treshold


    output = img.copy()

    for i in range(3, height - 3):
        for j in range(3, width - 3):
            count = 0
            curI = img_gray[i][j]
            activePixels = []

            count = 0
            for dx, dy in miniCircle:
                if isActivePixel(int(img_gray[i+dx][j+dy]), int(curI),
int(treshold)):
                    count += 1
            if count < 2:
                continue

            for dx,dy in bresenhamCircle:
                activePixels.append(isActivePixel(int(img_gray[i+dx][j+dy]),
int(curI), int(treshold)))

            activePixels_ext = activePixels + activePixels[:15]
            for num in activePixels_ext:
                if num == 1:
                    count += 1
                    if count == n:
                        output = cv.circle(output, (i, j), 2, (0, 0, 255), -1)
                else:
                    count = 0

    return output

img1 = FAST_Corners(img.copy(), n=10, treshold=15)
img2 = Harris_Corners(img)
cv.imshow('Image', np.hstack([img2, img1]))
cv.waitKey(0)
cv.destroyAllWindows()
```

Output:-

Date : 03/09/2025



2. Implement your own version of SIFT feature descriptors and compare with OpenCV library functions.

Assess the robustness of descriptors to changes in scale, rotation, and affine transformations. Also,

compare your implementation with the descriptors available in the opencv library. Use the earlier

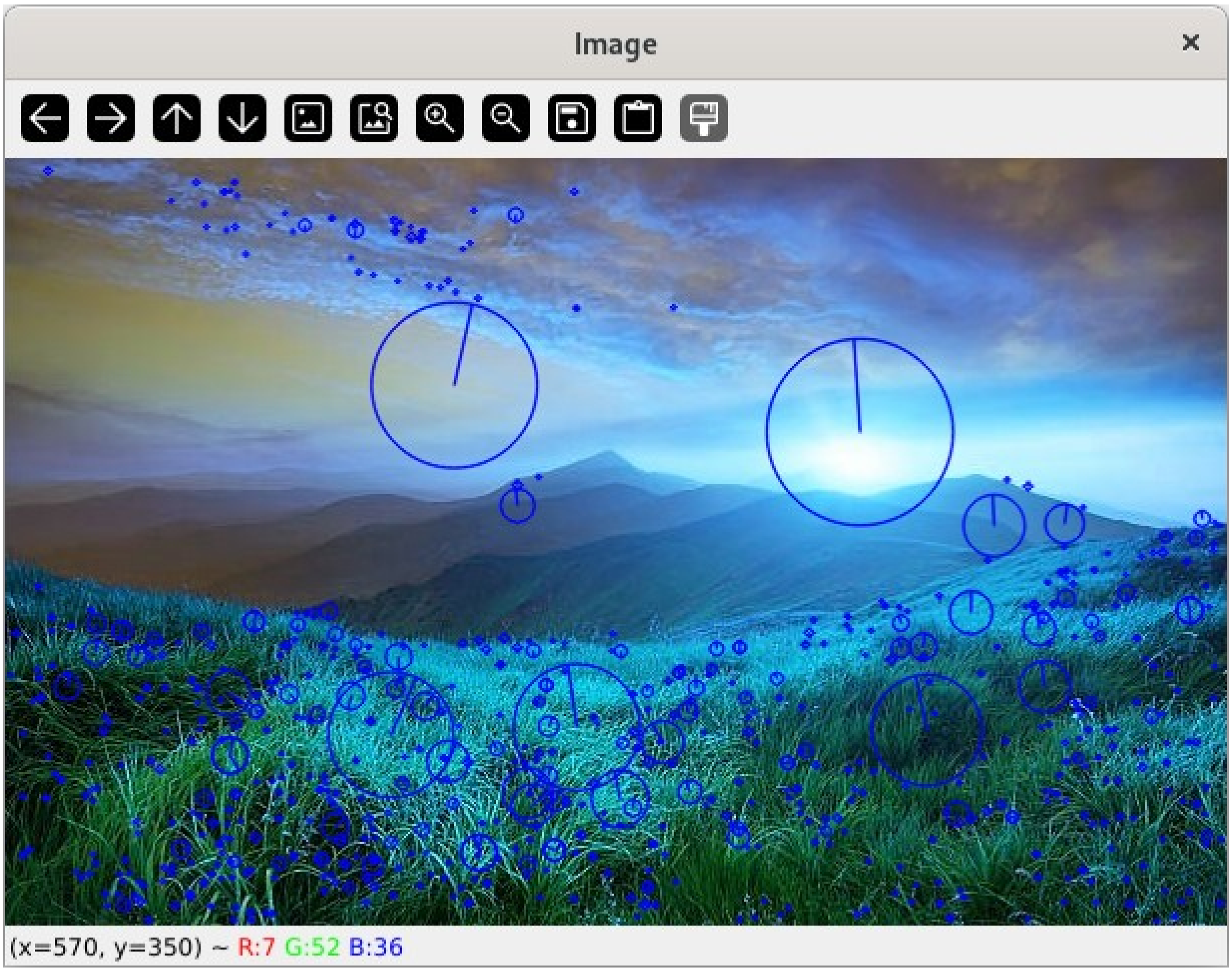
version of opencv (prior to OpenCV 3.4.3) for the SIFT.

Code:-

```
import cv2 as cv
import numpy as np

img = cv.imread('Week1/img.jpg')
img = cv.cvtColor(img, cv.COLOR_BGR2RGB)

sift = cv.SIFT_create()
keypoints, descriptors = sift.detectAndCompute(img, None)
keypoints_with_size = np.copy(img)

cv.drawKeypoints(img, keypoints, keypoints_with_size, color = (255, 0, 0),flags
= cv.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)

cv.imshow('Image', keypoints_with_size)
cv.waitKey(0)
```

Output:-

Date : 03/09/2025



3. Implement the HoG descriptor and apply it to detect humans in images. You may follow the steps given

below:

a) Obtain a dataset containing images with humans and non-human objects. Compute HoG for these images as references.

b) Extract HoG features using a sliding window approach.

c) Calculate a similarity score or distance metric between the HoG descriptor of the window and a reference HoG descriptor.

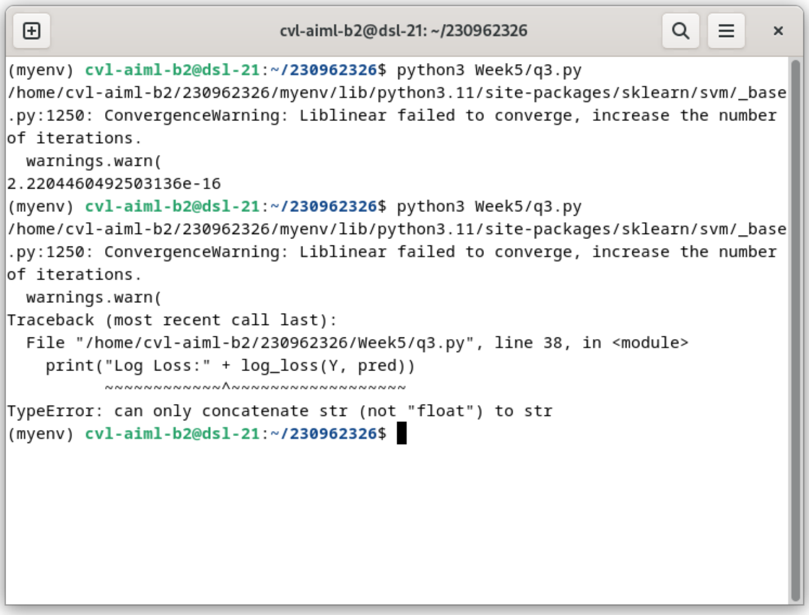
c) Identify and collect windows that exceed the similarity score threshold. Choose the best window among the overlapping windows.

Date : 03/09/2025

Code:-

```
import cv2 as cv
import numpy as np
from sklearn.svm import LinearSVC
from sklearn.metrics import log_loss

dataset_train = 'INRIAPerson/train_64x128_H96/'
dataset_test = 'INRIAPerson/test_64x128_H96/'

neg_list, pos_list =[], []

with open(dataset_train+'pos.lst', 'r') as f:
    pos_list = f.read().strip().split('\n')

with open(dataset_train+'neg.lst', 'r') as f:
    neg_list = f.read().strip().split('\n')

X, Y = [], []

hog = cv.HOGDescriptor((64, 128), (16, 16), (8, 8), (8, 8), 9)
for file_name in pos_list:
    img = cv.imread(dataset_train + file_name, 0)
    X.append(hog.compute(img))
    Y.append(1)

for file_name in neg_list[:300]:
    img = cv.imread(dataset_train + file_name, 0)
    img = cv.resize(img, (96, 160), interpolation=cv.INTER_CUBIC)

    X.append(hog.compute(img))
    Y.append(0)

X , Y = np.array(X), np.array(Y)

model = LinearSVC(max_iter=100)
model.fit(X, Y)

pred = model.predict(X)
print(log_loss(Y, pred))
```

Output:-



```
(myenv) cvl-aiml-b2@dsl-21:~/230962326$ python3 Week5/q3.py
/home/cvl-aiml-b2/230962326/myenv/lib/python3.11/site-packages/sklearn/svm/_base
.py:1250: ConvergenceWarning: Liblinear failed to converge, increase the number
of iterations.
  warnings.warn(
2.220446049250313 6e-16
(myenv) cvl-aiml-b2@dsl-21:~/230962326$ python3 Week5/q3.py
/home/cvl-aiml-b2/230962326/myenv/lib/python3.11/site-packages/sklearn/svm/_base
.py:1250: ConvergenceWarning: Liblinear failed to converge, increase the number
of iterations.
  warnings.warn(
Traceback (most recent call last):
  File "/home/cvl-aiml-b2/230962326/Week5/q3.py", line 38, in <module>
    print("Log Loss:" + log_loss(Y, pred))
          ~~~~~~~~~~~~^~~~~~~~~~~~~~~~~~~
TypeError: can only concatenate str (not "float") to str
(myenv) cvl-aiml-b2@dsl-21:~/230962326$
```