BansilalRamnath Agarwal Charitable Trust's

VISHWAKARMA INSTITUTE OF TECHNOLOGY

(An Autonomous Institute Affiliated to Savitribai Phule Pune University)

Bansilal Ramnath Agarwal Charitable Trust's

Vishwakarma Institute of Technology

Department of Electronics &Telecommunication Engineering

ET3021:DIGITAL IMAGE PROCESSNG

COURSE PROJECT REPORT

A.Y.2019-20 (SEM-II)

Title of Course project

Currency detection

Batch:B1

Name	Roll.no	GR.no
Diksha Avhad	3	182058
Ashutosh Badrapurkar	5	1710459
Smruti Khire	38	1710206
Aman Ladkat	45	1710679

T.Y. B.Tech.

Div: K

Guide: Prof. (Dr.)Ashutosh Marathe

INDEX

1. INTRODUCTION

There are approximately 50 currencies all over the world, with each of them looking totally different in terms of size, pattern and colour. An automatic currency identifier is the need of all financial institutions. It is important in every field where money related transactions are involved. For bank staffs, there is a "Currency Sorting Machine" helps them to recognize different kinds of currencies. Pattern recognition systems developed some techniques which are very helpful in the image matching. These technologies may help to develop automatic recognition systems that allow the image matching in real time, thus such systems are found to be reliable to implement in the day to day life.

In this project we have implemented a reliable currency recognition technique based on ORB Descriptor and BF Matcher.

2. SYSTEM DESCRIPTION AND WORKING

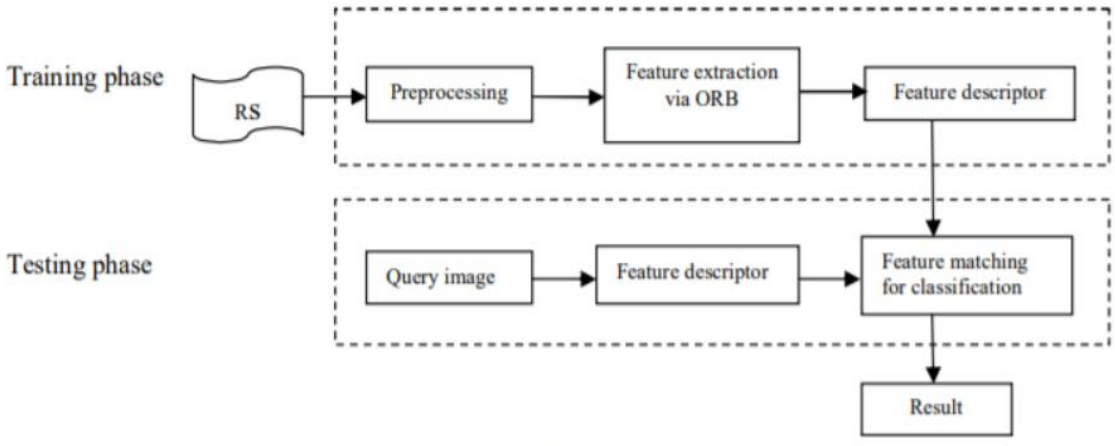


Fig1.Block Diagram of Currency Detection System

Overall there are nine types of Indian currency denominations (from front) considered. Denominations considered are Rs. 5, 10(coin and note), 20, 50, 100, 200,500, 2000.These all are included in the training dataset. The project is implemented with the help of Python in PyCharm IDE. The libraries used are: cv2, matplotlib, numpy.

It has two phases for the recognition purpose which are as follows:

1. **Training phase:**

   The training phase contains all pre-processing mechanism which extracts the information from the input image. Preprocessing initiates with image acquisition. Since there are nine denominations considered, all of these notes differ in size. Due to this, image resizing is done using ***resize_image()*** function.

   After that it goes for feature extraction via ORB where feature descriptors of that preprocessed image are calculated. This process is repeated for all samples of currency with different denominations.

   ORB is one method that is the development of the previous method, namely FAST and BRIEF so that it has advantages in speed of detection and resistance to rotation and noise FAST techniques are used in the initial stages to determine Keypoint. FAST does not calculate the orientation and rotation of the variant, but calculating the intensity of the centroid patch with its center angle, orientation is the direction of the vector from the midpoint to the centroid. Moments are calculated to increase rotation invariance. At ORB, the matrix rotation is calculated using the orientation of the patch and then the BRIEF descriptor directs it based on orientation. The ORB algorithm is one of the most efficient in image processing. ORB algorithm is an algorithm based on binary values. Where there are two main parts, namely FAST keypoint detector and BRIEF rotation which is enhanced in such a way that it can improve its ability to detect and describe features.

   To use ORB in OpenCV we have to create an ORB object with the function ***cv2.ORB()***. To compute the descriptor we use the ***ORB.detectAndCompute()*** function. The descriptor generated with ORB is bit string.

2. **Testing phase:**

   This phase consists of updation of the process. The test image is taken and allowed for preprocessing and feature extraction. When feature extraction is completed for test image, the feature vectors obtained are compared with that of feature vectors of original image samples. Feature matching is done by using Brute-Force matcher. It takes the descriptor of one feature in first set and is matched with all other features in second set using some distance calculation and the closest one is returned. For BF matcher, first we have to create

the BFMatcher object using ***cv2.BFMatcher()***. After that ***BFMatcher.knnMatch()*** method is used. This method returns k best matches where k is specified by the user. k=2 in our system.

CODE

```
from utils import *
from matplotlib import pyplot as plt
import os
import cv2

max_val = 8
max_pt = -1
max_kp = 0

def read_img(file_name):
        img = cv2.imread(file_name,cv2.IMREAD_UNCHANGED)
        return img

def resize_img(image, dim):
        res = cv2.resize(image,dim, interpolation = cv2.INTER_AREA)
        return res

print("Welcome To COMPUTER VISION FRAMEWORK FOR INDIAN CURRENCY RECOGNITION")
orb = cv2.ORB_create()
videoCaptureObject = cv2.VideoCapture(0)
result = True
while (result):
   ret, frame = videoCaptureObject.read()
   cv2.imwrite("D:/Currency/NewPicture.jpg", frame)
   result = False
videoCaptureObject.release()
cv2.destroyAllWindows()

test_img = read_img('D:/Currency/NewPicture.jpg')

scale_percent = 40
width = int(test_img.shape[1] * scale_percent / 100)
height = int(test_img.shape[0] * scale_percent / 100)
dim = (width, height)

original = resize_img(test_img, dim)
print(original)
```

```
(kp1, des1) = orb.detectAndCompute(test_img, None)

training_set = ['D:/Currency/5.jpg', 'D:/Currency/10.jpg',
                'D:/Currency/10_coin.jpg', 'D:/Currency/20.jpg',
                'D:/Currency/50.jpg', 'D:/Currency/100.jpg',
                'D:/Currency/200.jpg', 'D:/Currency/500.jpg',
                'D:/Currency/2000.jpg']

for i in range(0, len(training_set)):
    # train image
    train_img = cv2.imread(training_set[i])

    (kp2, des2) = orb.detectAndCompute(train_img, None)

    bf = cv2.BFMatcher()
    all_matches = bf.knnMatch(des1, des2, k=2)

    good = []
    for (m, n) in all_matches:
        if m.distance < 0.789 * n.distance:
            good.append([m])

    if len(good) > max_val:
        max_val = len(good)
        max_pt = i
        max_kp = kp2

    print(i, ' ', training_set[i], ' ', len(good))

if max_val != 8:
    # print(training_set[max_pt])
    print('good matches ', max_val)

    train_img = cv2.imread(training_set[max_pt])
    img3 = cv2.drawMatchesKnn(test_img, kp1, train_img, max_kp, good, 4)
    note = str(training_set[max_pt])[66:-4]
    print('\nDetected denomination: Rs. ', note)
    (plt.imshow(img3), plt.show())

else:
    print('No Matches')
```

RESULTS

1. **Rs 5**

Input image :

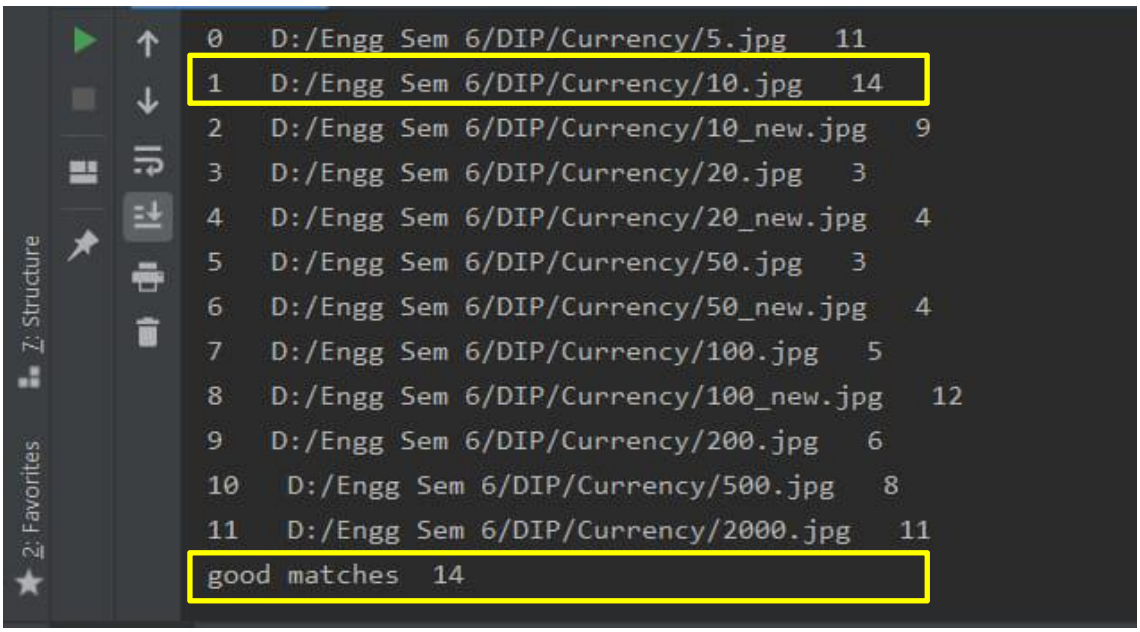
Output image :

The number of matches corresponding to each training image is shown. The yellow box is drawn to highlight that the highest number of matches (15) is found corresponding to Rs 5 image thus the correct currency is detected.

2. Rs 10 (Note)

Input image :

Output image :

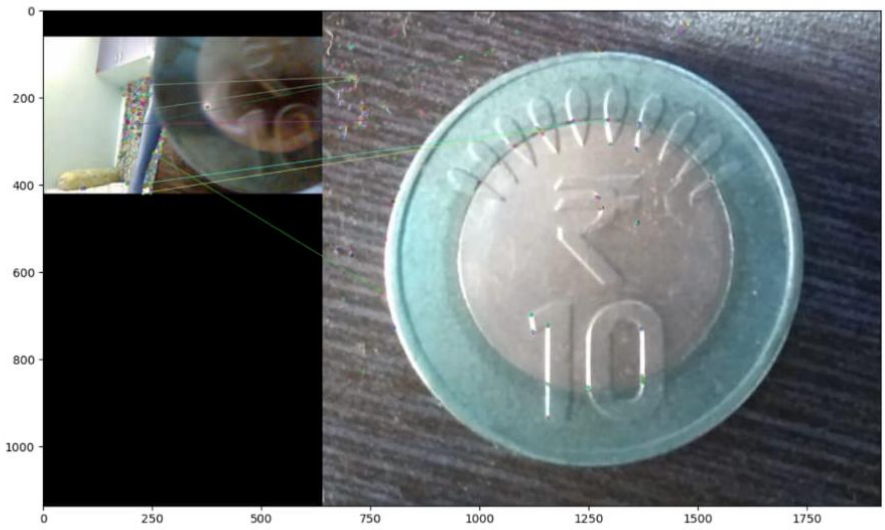


The number of matches corresponding to each training image is shown. The yellow box is drawn to highlight that the highest number of matches (14) is found corresponding to Rs 10 image thus the correct currency is detected.

3. **Rs 10(Coin)**

Input image :

Output image :

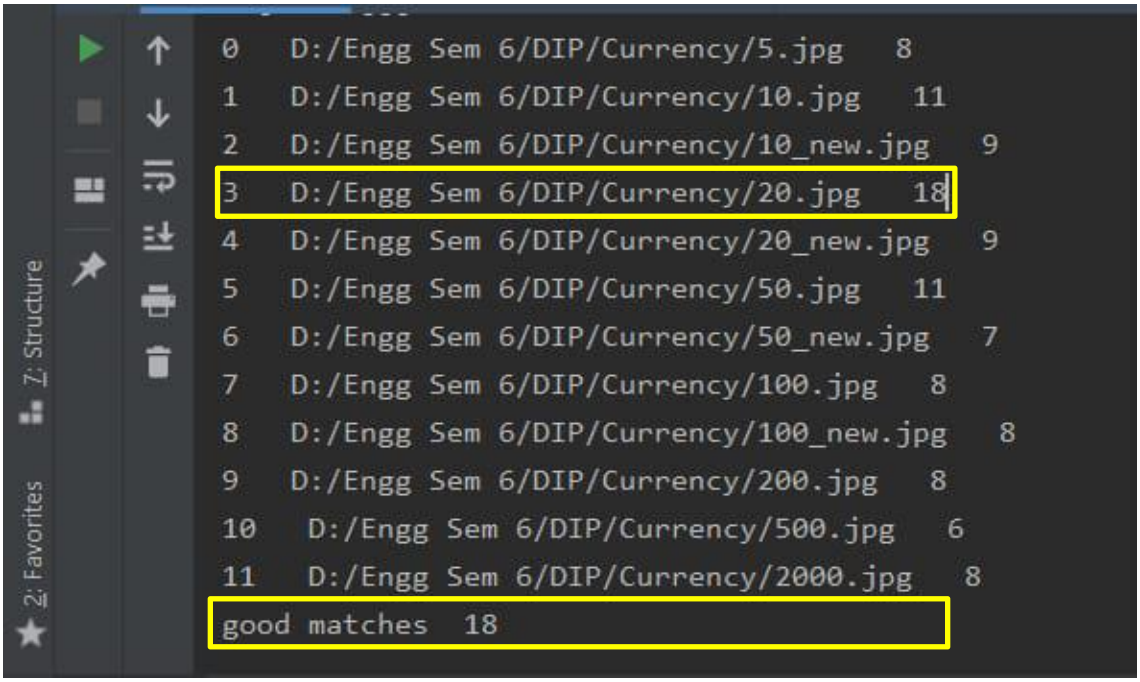


The number of matches corresponding to each training image is shown. The yellow box is drawn to highlight that the highest number of matches (12) is found corresponding to Rs 10 coin image thus the correct currency is detected.

4. **Rs 20**

Input image :

Output image :



The number of matches corresponding to each training image is shown. The yellow box is drawn to highlight that the highest number of matches (18) is found corresponding to Rs 20 image thus the correct currency is detected.

5. **Rs 50**

Input image :

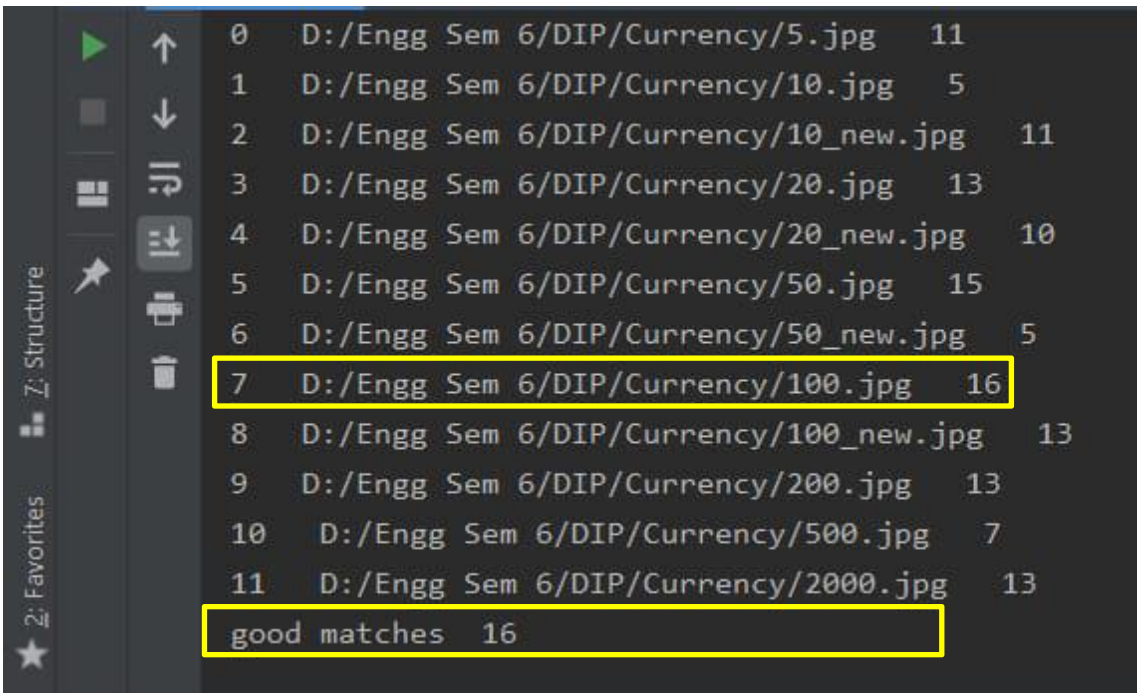
Output image :

The number of matches corresponding to each training image is shown. The yellow box is drawn to highlight that the highest number of matches (12) is found corresponding to Rs 50 image thus the correct currency is detected.

6. **Rs 100**

Input image :

Output image :



The number of matches corresponding to each training image is shown. The yellow box is drawn to highlight that the highest number of matches (16) is found corresponding to Rs 100 image thus the correct currency is detected.

7. **Rs 200**

Input image :

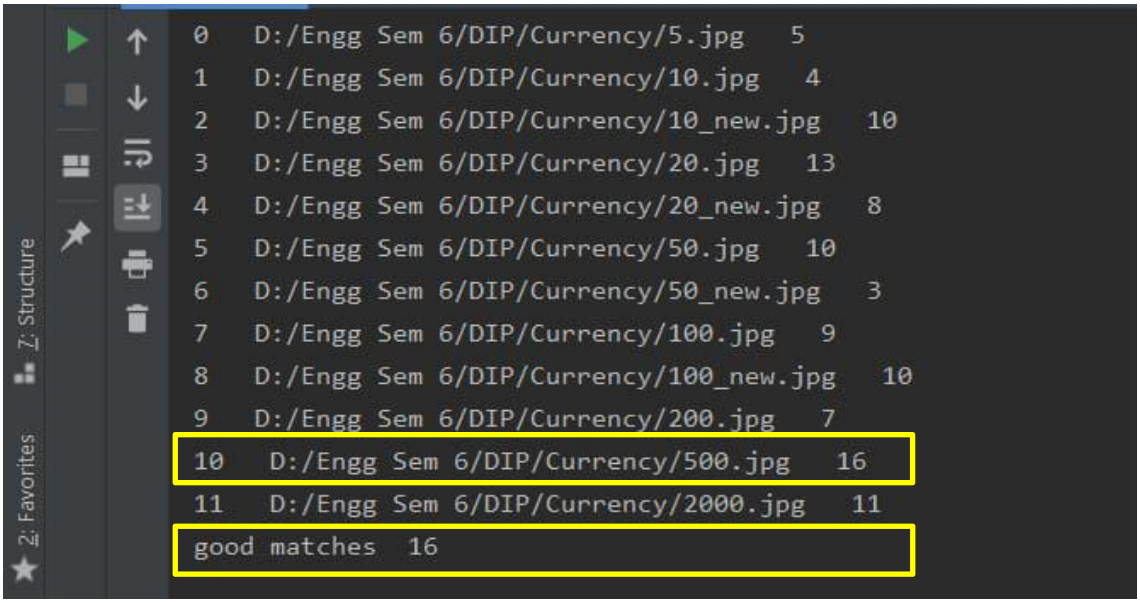
Output image :

The number of matches corresponding to each training image is shown. The yellow box is drawn to highlight that the highest number of matches (22) is found corresponding to Rs 200 image thus the correct currency is detected.

8. Rs 500

Input image :

Output image :

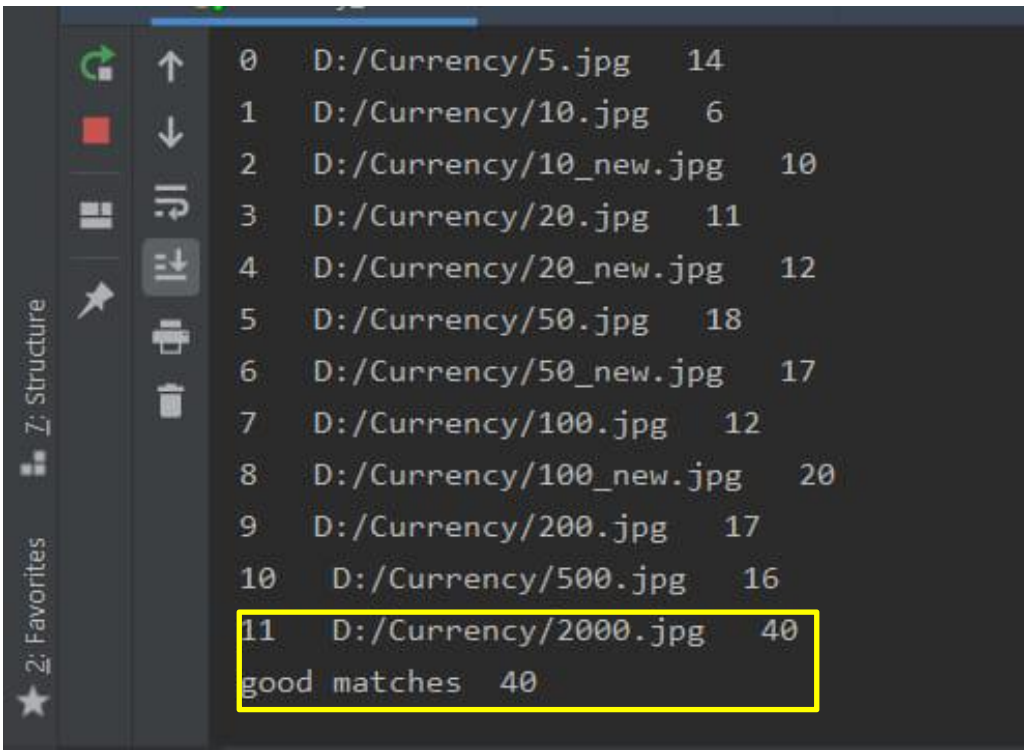


The number of matches corresponding to each training image is shown. The yellow box is drawn to highlight that the highest number of matches (16) is found corresponding to Rs 500 image thus the correct currency is detected.

9. Rs 2000

Input image :

Output image :



The number of matches corresponding to each training image is shown. The yellow box is drawn to highlight that the highest number of matches (40) is found corresponding to Rs 2000 image thus the correct currency is detected.

10. Random Input

Input image :

Output image :

The yellow box highlights that no matches are found for the random input image.

CONCLUSION

An innovative model for currency recognition system using digital image processing has been discussed. Techniques till date are useful but are complex, some of them are costly and also time consuming. The technique of Indian currency recognition based on ORB is much faster so that it can be helpful to increase computational speed. The note samples are considered from front side only hence the system needs to be trained to detect notes from the backside as well thus making it more reliable.

REFERENCES

[1] https://opencv-pythontutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_orb/py_orb.html [Accessed on: 21st April,2020]

[2]https://opencvpythontutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_matcher/py_matcher.html#goal [Accessed on: 21st April,2020]

[3] Nugraha, N. A., Irawan, B., & Prasasti, A. L. (2018). *Singapore Dollar Recognition Using ORB Feature Based on Android. 2018 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC).*