%pylab inline

import cv2

import numpy as np

from matplotlib import pyplot as plt

import matplotlib.image as mpimg

img\_rgb = cv2.imread('RainForest.png')

img\_gray = cv2.cvtColor(img\_rgb, cv2.COLOR\_BGR2GRAY)

template = cv2.imread('Rama.png',0)

w, h = template.shape[::-1]

res = cv2.matchTemplate(img\_gray,template,cv2.TM\_CCOEFF\_NORMED)

threshold = 0.8

loc = np.where( res >= threshold)

for pt in zip(\*loc[::-1]):

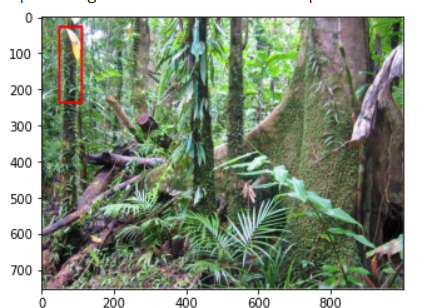
    cv2.rectangle(img\_rgb, pt, (pt[0] + w, pt[1] + h), (0,0,255), 2)

cv2.imwrite('res.png',img\_rgb)

img = mpimg.imread('res.png')

imgplot = plt.imshow(img)

plt.show()



%pylab inline

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

img = mpimg.imread('RainForest.png')

imgplot = plt.imshow(img)

plt.show()

import cv2

import numpy as np

from matplotlib import pyplot as plt

img\_rgb = cv2.imread('RainForest.png')

img\_gray = cv2.cvtColor(img\_rgb, cv2.COLOR\_BGR2GRAY)

template = cv2.imread('Rama.png',0)

w, h = template.shape[::-1]

res = cv2.matchTemplate(img\_gray,template,cv2.TM\_CCOEFF\_NORMED)

threshold = 0.8

loc = np.where( res >= threshold)

for pt in zip(\*loc[::-1]):

    cv2.rectangle(img\_rgb, pt, (pt[0] + w, pt[1] + h), (0,0,255), 2)

cv2.imwrite('res.png',img\_rgb)

True / False

import cv2

import numpy as np

from matplotlib import pyplot as plt

img = cv2.imread('RainForest.png',0)

img2 = img.copy()

template = cv2.imread('Rama.png',0)

w, h = template.shape[::-1]

# All the 6 methods for comparison in a list

methods = ['cv2.TM\_CCOEFF', 'cv2.TM\_CCOEFF\_NORMED', 'cv2.TM\_CCORR',

            'cv2.TM\_CCORR\_NORMED', 'cv2.TM\_SQDIFF', 'cv2.TM\_SQDIFF\_NORMED']

for meth in methods:

    img = img2.copy()

    method = eval(meth)

    # Apply template Matching

    res = cv2.matchTemplate(img,template,method)

    min\_val, max\_val, min\_loc, max\_loc = cv2.minMaxLoc(res)

    # If the method is TM\_SQDIFF or TM\_SQDIFF\_NORMED, take minimum

    if method in [cv2.TM\_SQDIFF, cv2.TM\_SQDIFF\_NORMED]:

        top\_left = min\_loc

    else:

        top\_left = max\_loc

    bottom\_right = (top\_left[0] + w, top\_left[1] + h)

    cv2.rectangle(img,top\_left, bottom\_right, 255, 2)

    plt.subplot(121),plt.imshow(res,cmap = 'gray')

    plt.title('Matching Result'), plt.xticks([]), plt.yticks([])

    plt.subplot(122),plt.imshow(img,cmap = 'gray')

    plt.title('Detected Point'), plt.xticks([]), plt.yticks([])

    plt.suptitle(meth)

    plt.show()