**Лабораторная работа №6**

# Соответствие Шаблону

*Цели*

*В этой главе вы узнаете:*

*Поиск объектов на изображении с помощью сопоставления шаблонов*

*Вы увидите следующие функции : cv2.matchTemplate(), cv2.minMaxLoc()*

## cv2.matchTemplate()

Сопоставление шаблонов-это метод поиска и нахождения местоположения шаблонного изображения на более крупном изображении. OpenCV поставляется с функцией **cv2.matchTemplate()** для этой цели. Он просто скользит шаблонным изображением по входному изображению (как в 2D-свертке) и сравнивает шаблон и патч входного изображения под шаблонным изображением. В OpenCV реализовано несколько методов сравнения. (Вы можете проверить документы для получения более подробной информации). Он возвращает изображение в оттенках серого, где каждый пиксель обозначает, насколько соседство этого пикселя совпадает с шаблоном.

Если входное изображение имеет размер (WxH) и шаблонное изображение имеет размер (wxh), то выходное изображение будет иметь размер (W-w+1, H-h+1). Как только вы получили результат, вы можете использовать функцию **cv2.minMaxLoc ()**, чтобы найти, где находится максимальное/минимальное значение. Возьмите его как верхний левый угол прямоугольника и возьмите (w,h) как ширину и высоту прямоугольника. Этот прямоугольник - ваша область шаблона.

Если вы используете cv2.TM\_SQDIFF в качестве метода сравнения, минимальное значение дает наилучшее совпадение.





**import cv2**

**import numpy as np**

**from matplotlib import pyplot as plt**

**img = cv2.imread('12.jpg',0)**

**img2 = img.copy()**

**template = cv2.imread('12\_cat2.jpg',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')**

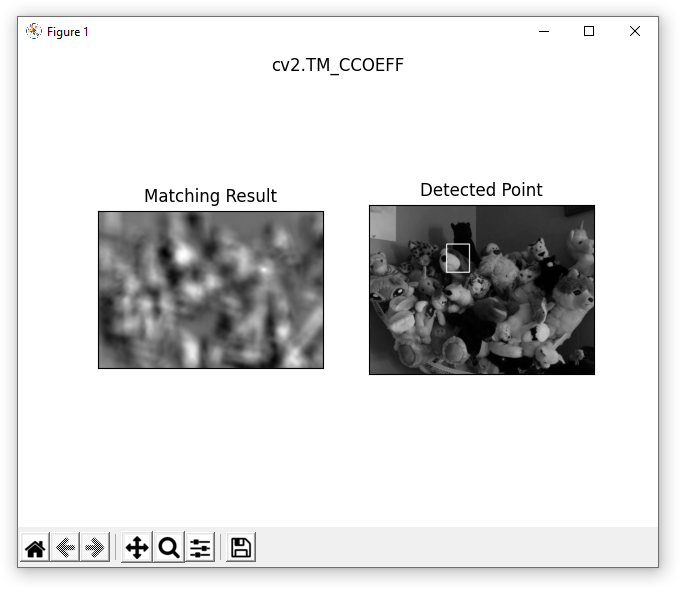
**plt.title('Matching Result'), plt.xticks([]), plt.yticks([])**

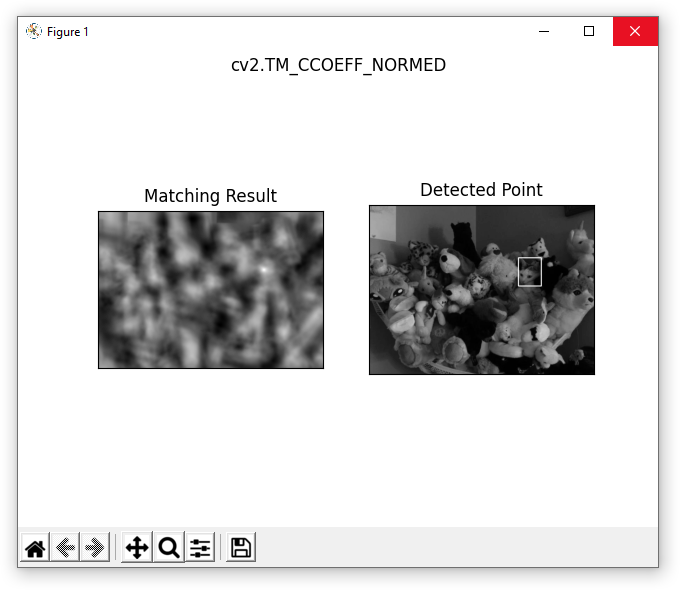
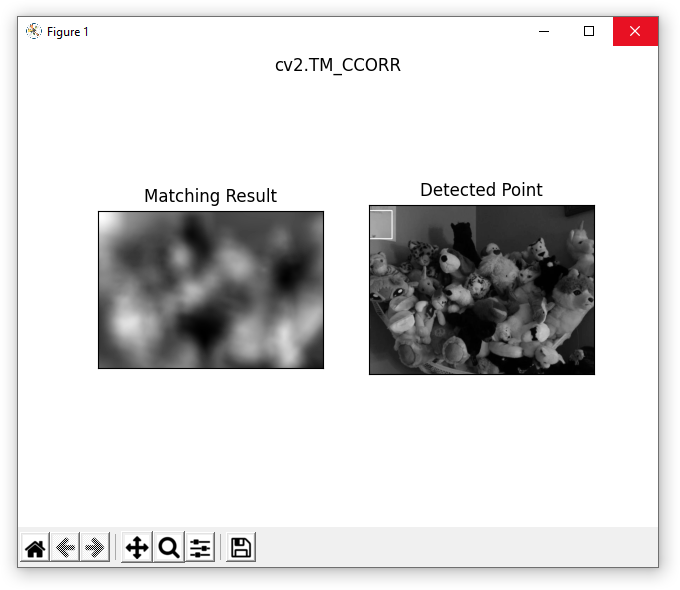
**plt.subplot(122),plt.imshow(img,cmap = 'gray')**

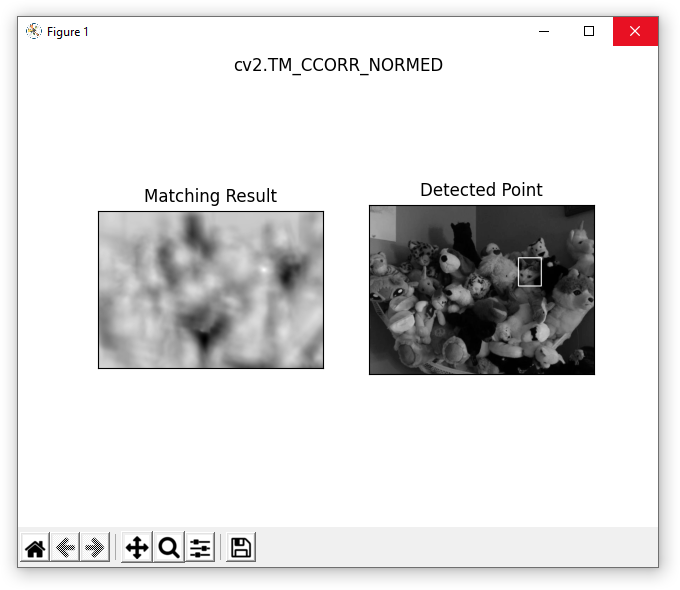
**plt.title('Detected Point'), plt.xticks([]), plt.yticks([])**

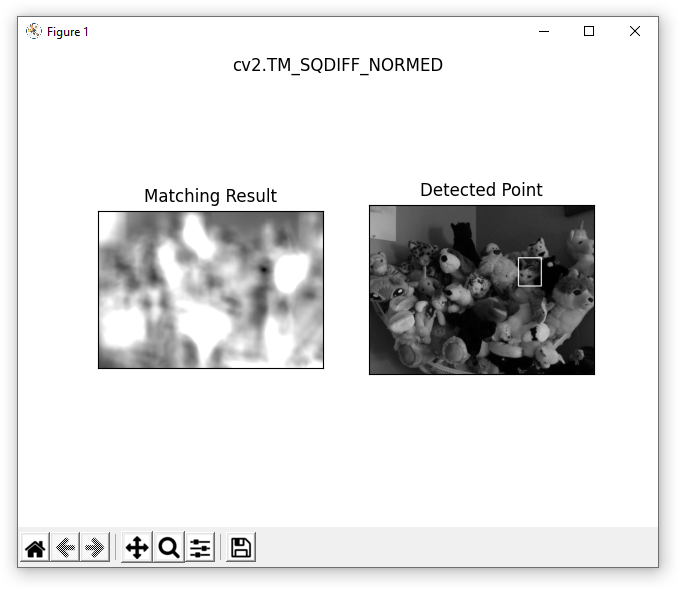
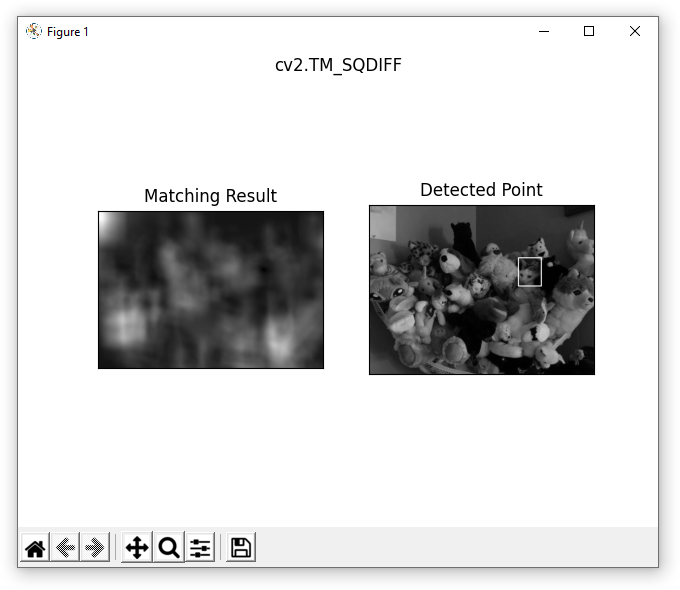
**plt.suptitle(meth)**

**plt.show()**



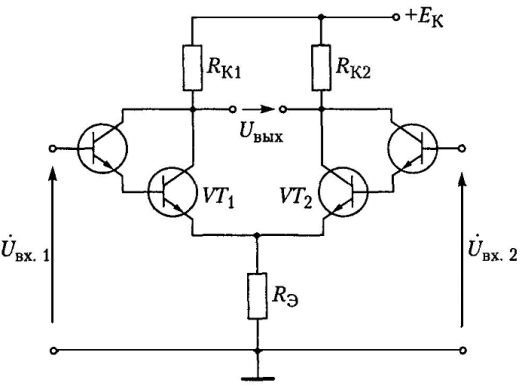
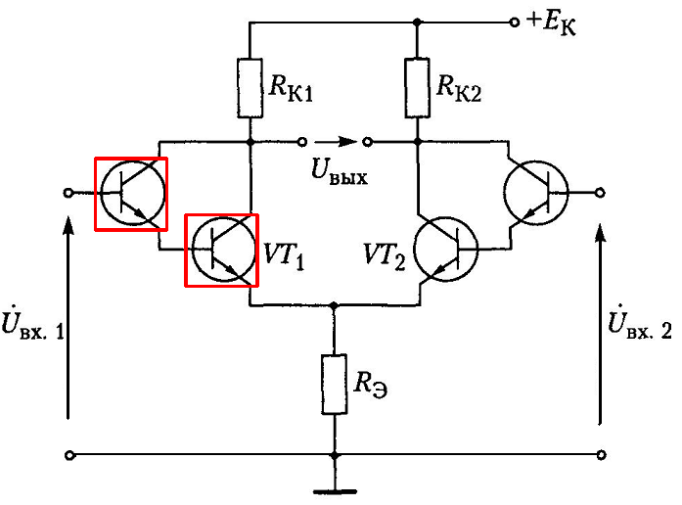






## Сопоставление шаблонов с несколькими объектами

В предыдущем разделе мы искали изображение лица Месси, которое встречается на изображении только один раз. Предположим, вы ищете объект, который имеет несколько вхождений, cv2.minMaxLoc() не даст вам всех местоположений. В этом случае мы будем использовать пороговое значение. Итак, в этом примере мы будем использовать скриншот знаменитой игры Mario и найдем в ней монеты.

**import cv2**

**import numpy as np**

**from matplotlib import pyplot as plt**

**img\_rgb = cv2.imread('sch.png')**

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

**template = cv2.imread('sch\_tr.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)**

Вопросы к лабораторной работе

1. привести подробное описание основных функций, рассмотренных в работе - функция, что делает, аргументы, параметры, возвращаемые результаты, типы данных и тп.
2. выполнить примеры, описанные в лабораторной - предоставить комментированный код, поэтапные результаты выполнения действий (исходные фото-, видео- данные для примеров берутся собственные, или из открытых источников или из базового набора библиотеки OpenCV)