

Template Matching

Search for the `t` using `./t_character.png` as template in the text image `./text_image.png`

Use a **bounding box** to mark where `t` were found. Use the **Euclidean norm**.

You may use *OpenCV* to only read and write the image, but not to call the template matching routine.

```
In [ ]: import cv2
import matplotlib.pyplot as plt
import numpy as np

In [ ]: def read_image(filename):
    img = cv2.imread(filename, cv2.IMREAD_GRAYSCALE)
    if img is None:
        print("Error opening image: " + filename)
        exit()
    print("Image size: ", img.shape)
    return img

In [ ]: def euclidean_norm(template, target, x, y, template_width, template_height):
    return (template - target[y:y+template_height, x:x+template_width])**2

In [ ]: # draw bounding box around the template
def bounding_box(target, template_width, template_height, x, y):
    cv2.rectangle(target, (x, y), (x + template_width,
        y + template_height), (0, 255, 0), 0)

In [ ]: # import the image and template as grayscale
target = read_image("./text_image.png")
template = read_image("./t_character.png")
target_height, target_width = target.shape
template_height, template_width = template.shape

Image size: (851, 634)
Image size: (9, 8)

In [ ]: # find the template in the target image
for x in range(0, target_width - template_width):
    for y in range(0, target_height - template_height):
        if np.sum(euclidean_norm(template, target, x, y, template_width, template_height)) == 0:
            bounding_box(target, template_width, template_height, x, y)

In [ ]: # write the image named "output.png"
cv2.imwrite("output.png", target)

# read output image and display it larger
output = cv2.imread("output.png")
plt.figure(figsize=(10, 10))
plt.imshow(output)
plt.show()
```

```
0 import numpy as np
import cv2 as cv2
import matplotlib.pyplot as plt

100 # In[9]:

def myImshow(title, img):
    """
    function to make windows display work in jupyter notebook
    - shows image in a separate window,
    - waits for any key to close the window.
    """

    cv2.startWindowThread()
    cv2.imshow(title, img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

200

300

400 # In[10]:

path = "D:/data/Dropbox/ML/"
#RGB images in BGR order in penCV
img1 = cv2.imread(path+'box.png',cv2.IMREAD_GRAYSCALE) # queryImage
# Print error message if image is null
if img1 is None:
    print('Could not read query image')
else:
    print("Query Image read success...")

500

600 img2 = cv2.imread(path+'box_in_scene.png',cv2.IMREAD_GRAYSCALE) # TargetImage
# Print error message if image is null
if img2 is None:
    print('Could not read Training image')
else:
    print("Target Image read success...")

700

# In[11]:

# Initialize SIFT detector
sift = cv2.SIFT_create()

800
```