Morphological operator implementation using CUDA

1106022 陳柏嘉

```
# 讀取圖片
image = cv2.imread('img/lahn.jpg',0)
```

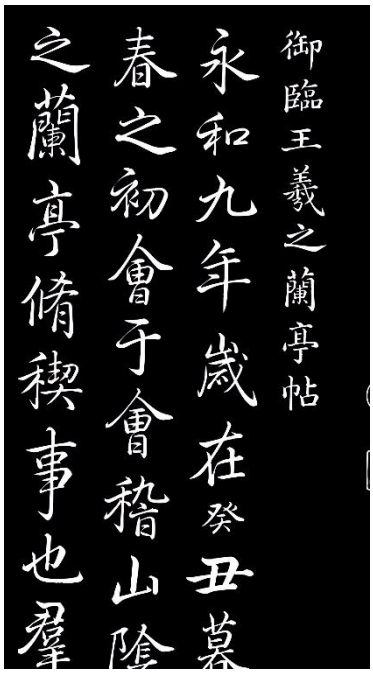
```
# 確認是否支持 GPU 運算
print('OpenCL available:', cv2.ocl.haveOpenCL())
```

```
output:
    OpenCL available: True
```

```
# set kernel size
kernel = np.ones((5,5), np.uint8)
```

將圖片放入 GPU 運算容器中 (UMat):

```
img_GPU = cv2.UMat(image)
```

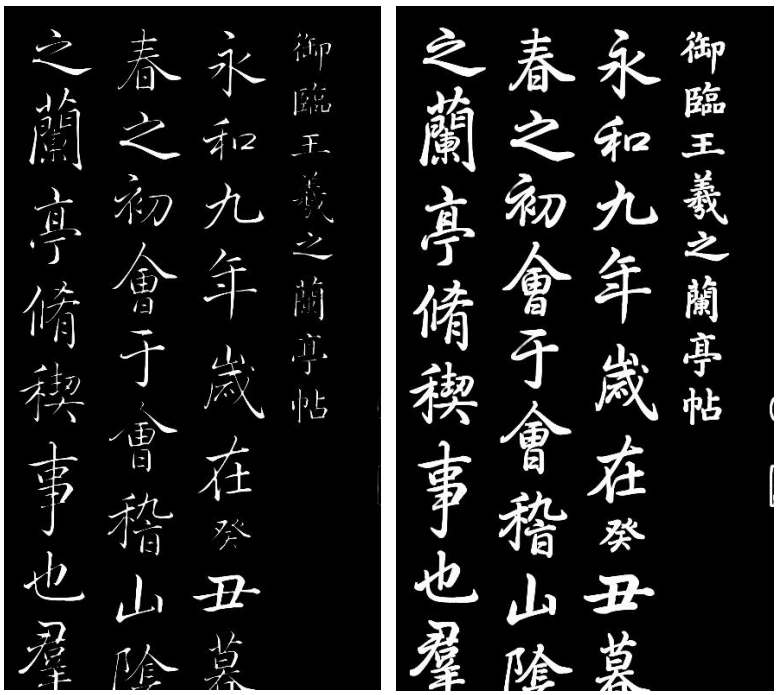


Origina

進行 Erosion 與 Dilation 運算:

```
## GPU ##
erosion = cv2.erode(img_GPU, kernel, iterations = 1)
dilation = cv2.dilate(img_GPU, kernel, iterations = 1)
```

Result:



(左) Erosion
(右) Dilation

Python Code：

```python
import cv2
import numpy as np


# 讀取圖片
image = cv2.imread('img/lahn.jpg',0)

# 確認是否支持 GPU 運算
print('OpenCL available:', cv2.ocl.haveOpenCL())

img_GPU = cv2.UMat(image)

# set kernel size
kernel = np.ones((5,5), np.uint8)

## CPU ##
# erosion = cv2.erode(img, kernel, iterations = 1)
# dilation = cv2.dilate(img, kernel, iterations = 1)

## GPU ##
erosion = cv2.erode(img_GPU, kernel, iterations = 1)
dilation = cv2.dilate(img_GPU, kernel, iterations = 1)

print(type(erosion))

cv2.imshow('Original', image)
cv2.imshow('Erosion', erosion)
cv2.imshow('Dilation', dilation)

cv2.imwrite("3-1_erosion.jpg", erosion)
cv2.imwrite("3-2_dilation.jpg", dilation)

cv2.waitKey(0)
```