# Computer Vision - Assignment 4

R09922A04 資工所人工智慧組 黃品硯

## (a) Dilation

Iteratively scan through the image's pixel, if the pixel is white (=255) then fill the white to the surrounding pixels by kernel. Union all of the results.

**[Output]**

**[Code]**

```
kernel = [(-2, -1), (-2, 0), (-2, 1), (-1, -2), (-1, -1), (-1, 0), (-1, 1),
(-1, 2), (0, -2), (0, -1), (0, 0), (0, 1), (0, 2), (1, -2), (1, -1), (1, 0),
(1, 1), (1, 2), (2, -1), (2, 0), (2, 1)]

for y in range(height):
  for x in range(width):
    if img[y, x] == 255:
      for rel_y, rel_x in kernel:
        new_y = y + rel_y
        new_x = x + rel_x
        if new_y < height and new_y >= 0 and new_x < width and new_x >= 0:
            out_img[new_y, new_x] = 255
```

## (b) Erosion

Iteratively scan through the image's pixel, take the pixel as the center of the kernel, if all of the surrounding pixels match the kernel then set the pixel to white, else set it to black.

**[Output]**

**[Code]**

```
if (0, 0) in kernel:
    target_pixel = 255
else:
    target_pixel = 0

for y in range(height):
    for x in range(width):
        if img[y, x] == target_pixel:
            check = True
            for rel_y, rel_x in kernel:
                check_y = y + rel_y
                check_x = x + rel_x
                if (
                    check_y < height
                    and check_y >= 0
                    and check_x < width
                    and check_x >= 0
                    and img[check_y, check_x] == 0
                ):
                    check = False
                    break

            if check:
                out_img[y, x] = 255
```

## (c) Opening

Do erosion, then dilation.

**[Output]**

**[Code]**

```
erosion_img = erosion(img, kernel)
out_img = dilation(erosion_img, kernel)
```

## (d) Closing

Do dilation, then erosion.

**[Output]**

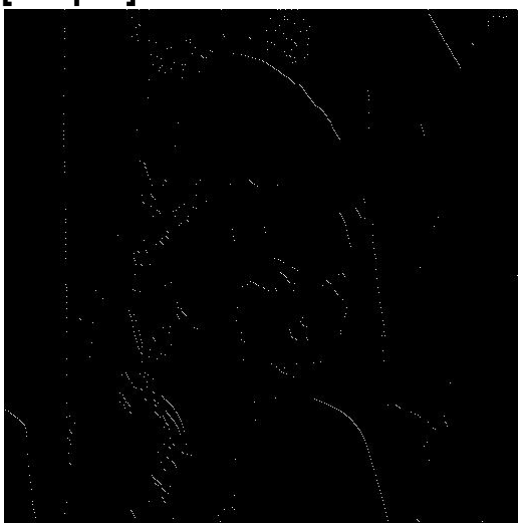
**[Code]**

```
dilation_img = dilation(img, kernel)
out_img = erosion(dilation_img, kernel)
```

## (e) Hit-and-miss transform

Do erosion with the original image and the kernel_j and do erosion with the complement of the original image and the kernel_k. Take the intersection of the two results and get the final output.

**[Output]**



**[Code]**

```
kernel_j = [(0, -1), (0, 0), (1, 0)]
kernel_k = [(-1, 0), (-1, 1), (0, 1)]

complement_img[img == 255] = 0
complement_img[img == 0] = 255

erosion_a_j = erosion(img.copy(), kernel_j)
erosion_ac_k = erosion(complement_img.copy(), kernel_k)

for y in range(height):
    for x in range(width):
        if erosion_a_j[y, x] == erosion_ac_k[y, x] == 255:
            out_img[y, x] = 255
```