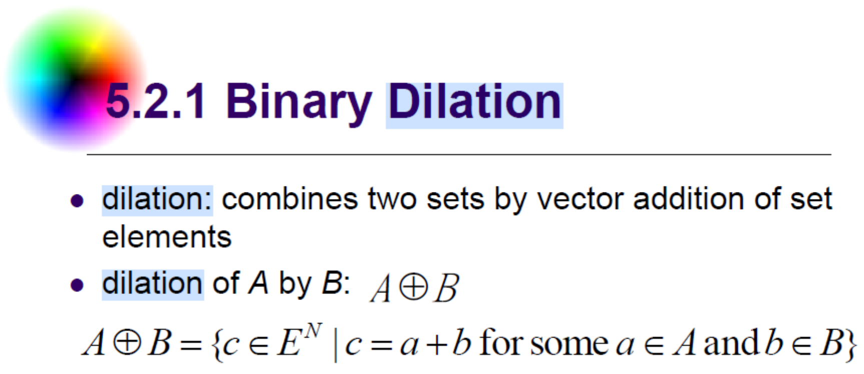
Computer Version HW4

Write programs which do binary morphology on a binary image:

	*	*	*	
*	*	*	*	*
*	*	*	*	*
*	*	*	*	*
	*	*	*	

(a) Dilation

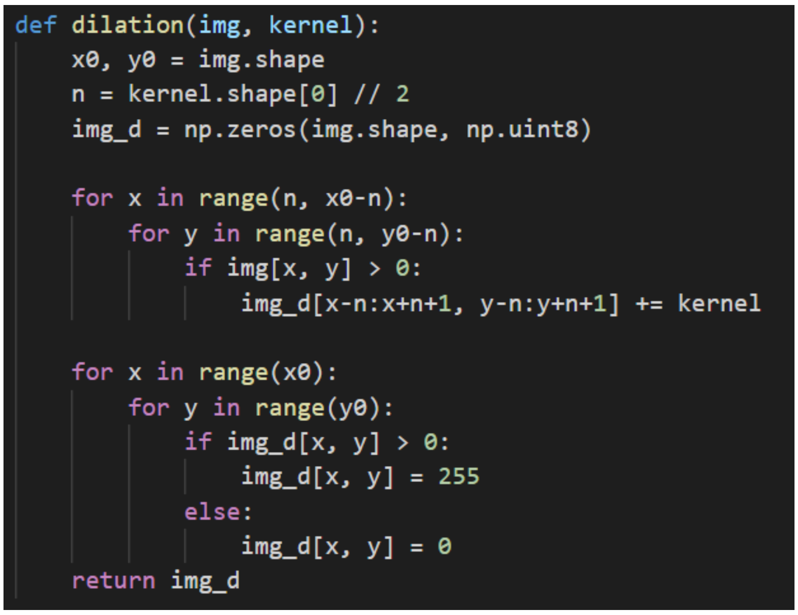
Combines 2 sets by vector addition of set elements



5.2.1 Binary Dilation

- dilation: combines two sets by vector addition of set elements
- dilation of *A* by *B*: $A \oplus B$

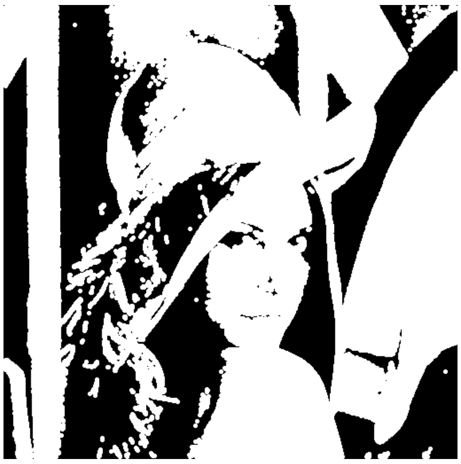
$$A \oplus B = \{c \in E^N \mid c = a + b \text{ for some } a \in A \text{ and } b \in B\}$$



```python
def dilation(img, kernel):
    x0, y0 = img.shape
    n = kernel.shape[0] // 2
    img_d = np.zeros(img.shape, np.uint8)

    for x in range(n, x0-n):
        for y in range(n, y0-n):
            if img[x, y] > 0:
                img_d[x-n:x+n+1, y-n:y+n+1] += kernel

    for x in range(x0):
        for y in range(y0):
            if img_d[x, y] > 0:
                img_d[x, y] = 255
            else:
                img_d[x, y] = 0
    return img_d
```



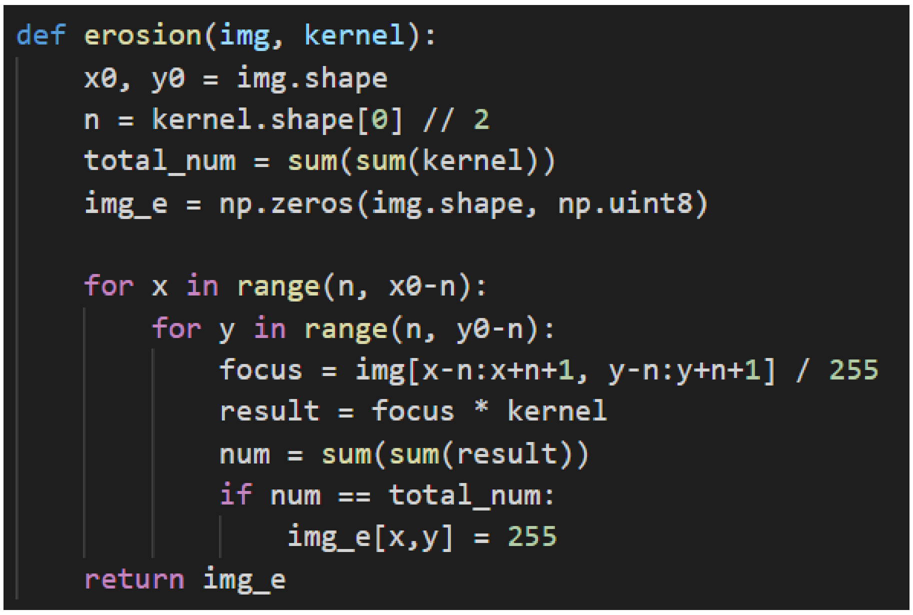
(b) Erosion

Morphological dual of dilation

5.2.2 Binary Erosion

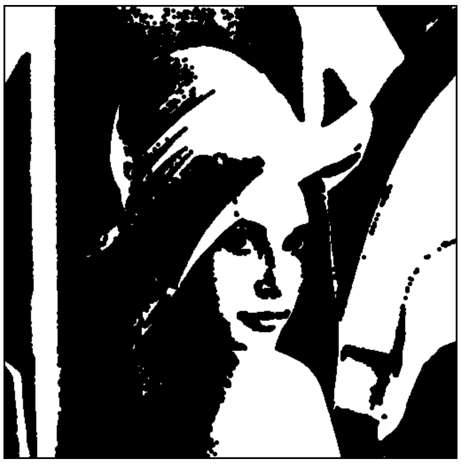
- erosion: morphological dual of dilation
- erosion of *A* by *B:* set of all *x* s.t. $x + b \in A$ for every $b \in B$

$$A \ominus B = \{x \in E^N | x + b \in A \text{ for every } b \in B\}$$



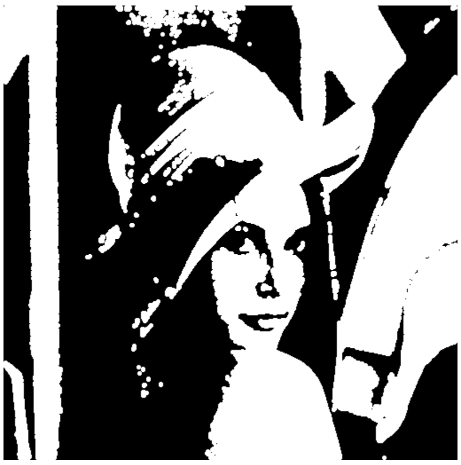
```python
def erosion(img, kernel):
    x0, y0 = img.shape
    n = kernel.shape[0] // 2
    total_num = sum(sum(kernel))
    img_e = np.zeros(img.shape, np.uint8)

    for x in range(n, x0-n):
        for y in range(n, y0-n):
            focus = img[x-n:x+n+1, y-n:y+n+1] / 255
            result = focus * kernel
            num = sum(sum(result))
            if num == total_num:
                img_e[x,y] = 255
    return img_e
```



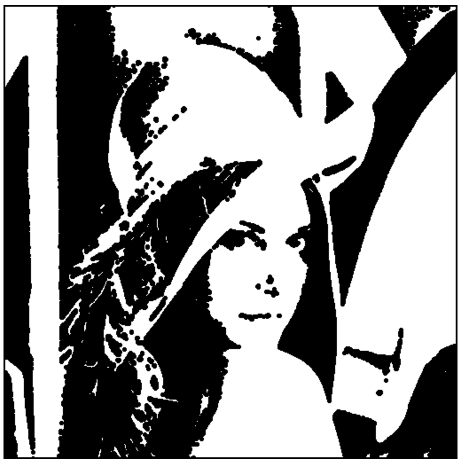
(c) Opening

先 erosion 再 dilation

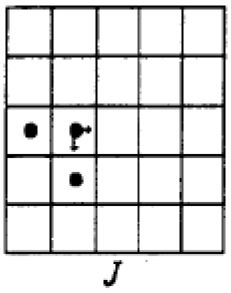


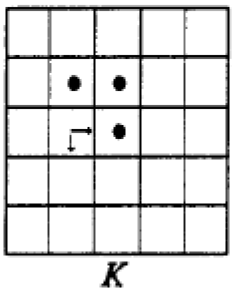
(d) Closing

先 dilation 再 erosion



(e) Hit-and-miss transform





- hit-and-miss of set *A* by (*J*,*K*)

$$A \otimes (J, K) = (A \ominus J) \cap (A^c \ominus K)$$