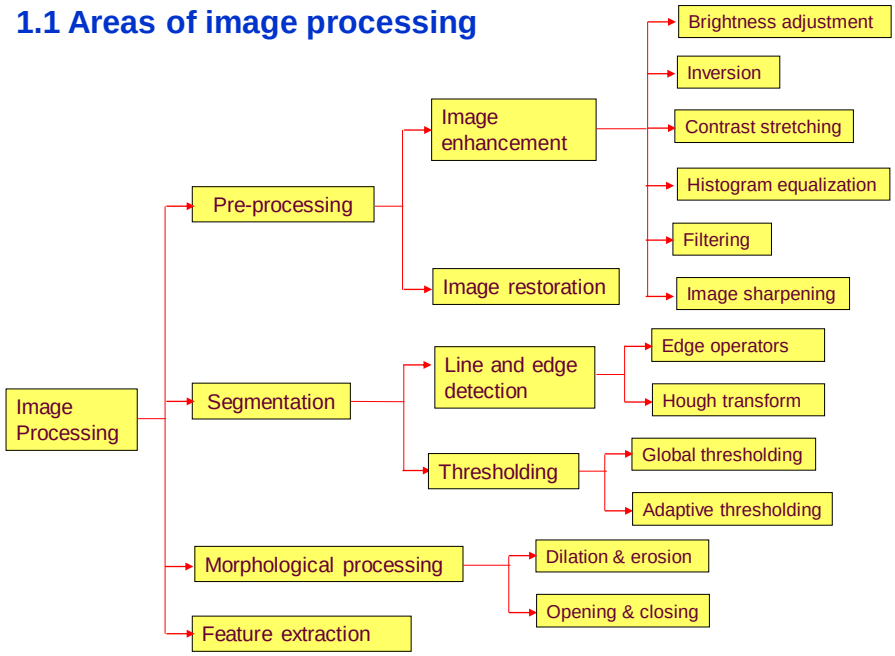
1.1 Areas of image processing



Before we begin

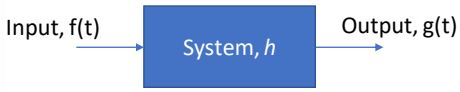
```
# read_image1.py - read and display image

import cv2 as cv

img = cv.imread('coin.jpg')
cv.imshow('image',img)

cv.waitKey(0)
cv.destroyAllWindows()
```

Before we begin

```
# smooth1.py - apply average filter

import cv2

I1 = cv2.imread('coin.jpg')
cv2.imshow('Coin',I1)

I2 = cv2.blur(I1,(5,5))
cv2.imshow('Coin - average filtered',I2)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Input, *f*

Output, *g*



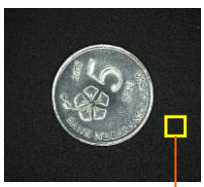
$$g(t) = \int_{-\infty}^{\infty} f(t)h(t-\tau)\mathrm{d}\tau$$

```
I2 = cv2.blur(I1,(5,5))
```

$$g = f * h$$

Convolution operation

1.2 Digital convolution and its application



$$g(i) = f(i) * h(i) = \sum_{k=-\infty}^{+\infty} f(k)h(i-k)$$

56	32	24	28	30
56	26	41	40	42
60	45	**53**	32	56
56	32	56	32	80
58	30	38	64	97
63	14	20	22	96

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

?	?	?	?	?
?	?	?	?	?
?	?	**?**	?	?
?	?	?	?	?
				?
				?

Output pixel value = (26x1/9) + (41x1/9) + …+ (56x1/9) + (32x1/9) = ??

• One-dimension digital convolution:

$$g(i) = f(i) * h(i) = \sum_{k=-\infty}^{+\infty} f(k)h(i-k)$$

• One-dimensional digital convolution over finite range, e.g. from $-w$ to $+w$

$$g(i) = f(i) * h(i) = \sum_{k=i-w}^{i+w} f(k)h(i-k)$$

• Two-dimensional digital convolution over finite range,

$$g(x,y) = f(x,y) * h(x,y) = \sum_i \sum_j f(i,j)h(x-i,y-j)$$

Point spread function (window/ mask/template)

• Digital convolution operation for a 3×3 point spread function

-1	0	1
-1	0	1
-1	0	1

h(x,y)

21	18	26	28	34	30	26
12	20	25	32	38	46	64
29	32	34	46	48	58	61
18	21	23	29	35	45	52
26	29	36	37	45	50	66
35	40	48	52	68	87	94
42	38	52	56	63	68	80

f(x,y)

					?	

g(x,y)

Output pixel value =
(-1×35)+(0×45)+(1×52)
+(-1×45)+(0×50)+(1×66)
+(-1×68)+(0×87)+(1×94) = 64