Welcome to

Computer Vision Week #2 - Lecture

AJ R&D

Agenda

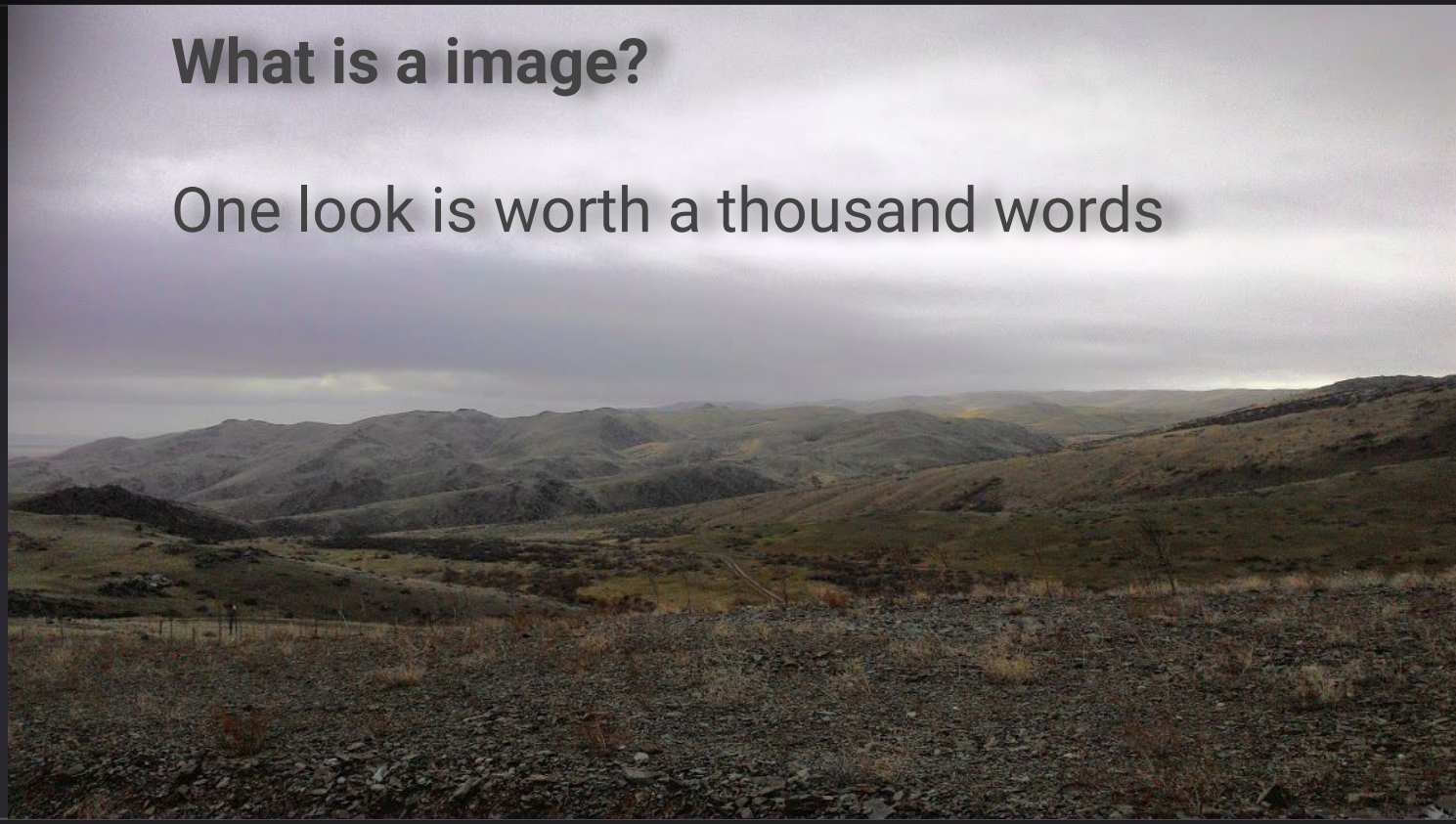
Week #2 - Computer Vision - Lecture
Image Processing for Computer Vision

1. Linear Image processing: Images as function
2. Linear Image processing: Filtering

Agenda

Week #2 - Computer Vision - Lecture
Image Processing for Computer Vision

1. Linear Image processing: Images as function
2. Linear Image processing: Filtering


What is a image?
One look is worth a thousand words

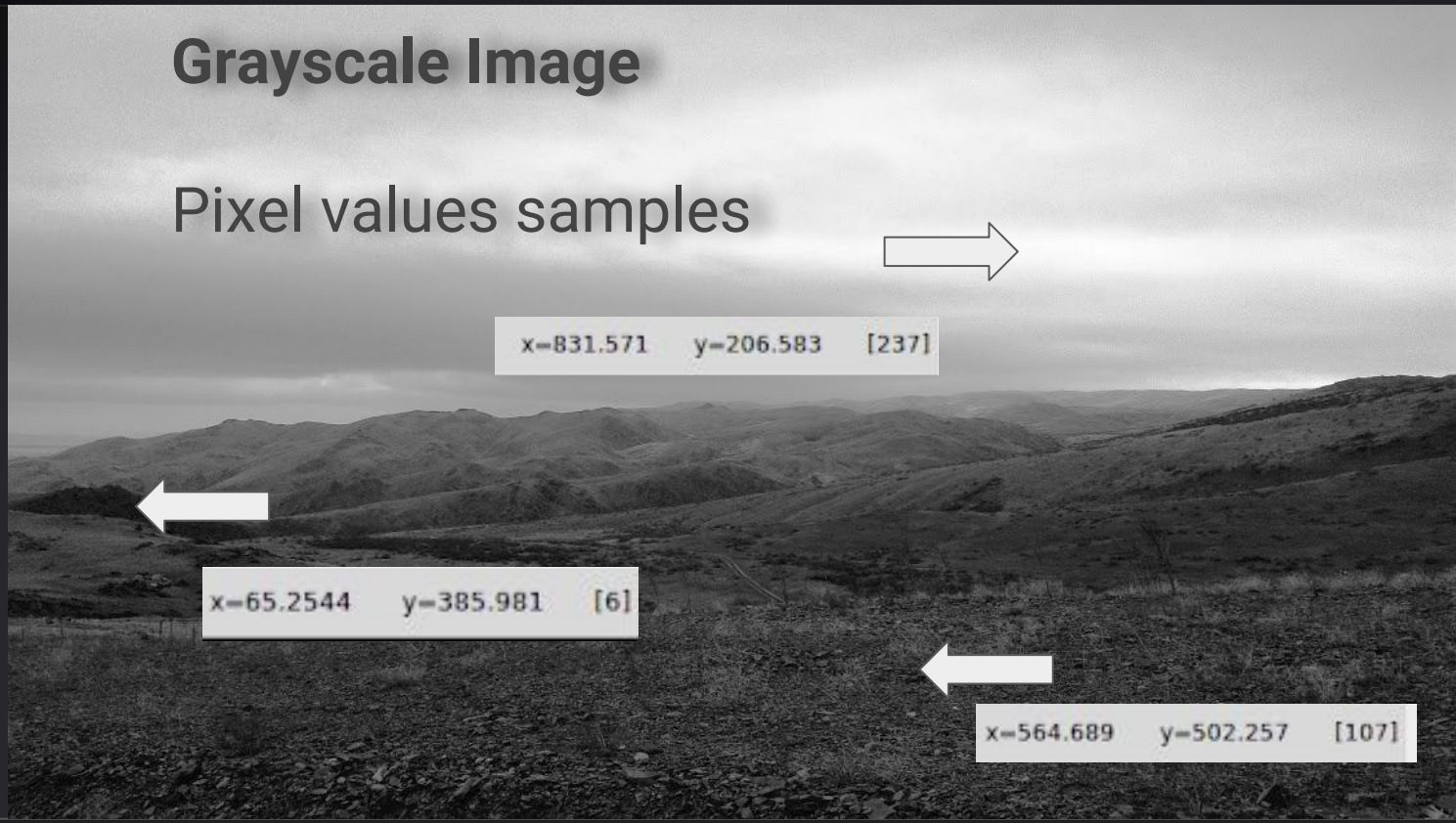
What is a image?

A (color) image is a 3D tensor of numbers

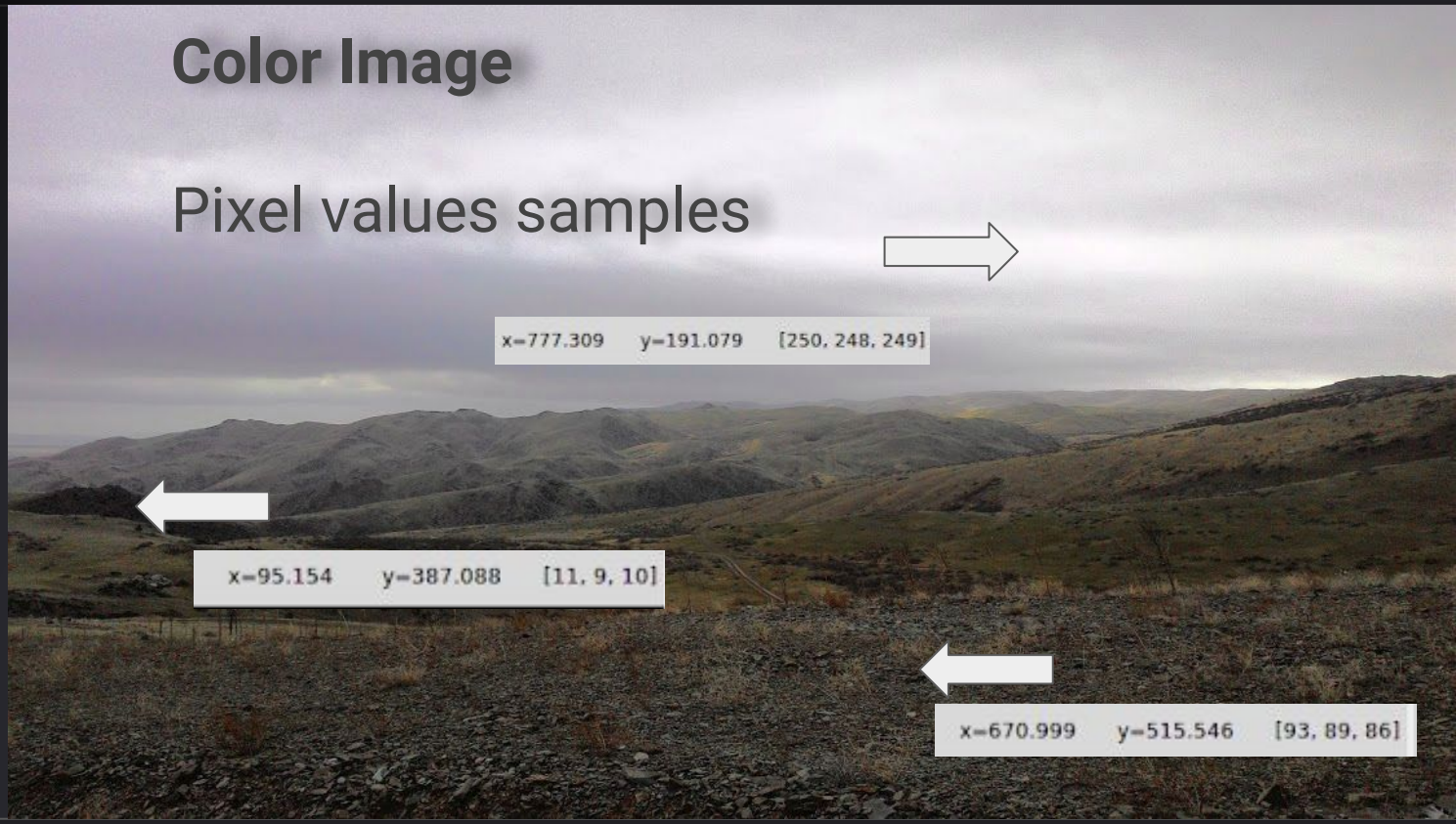
Combination of 3 channels

Red Green Blue


Grayscale Image
An Image contains discrete number of pixels
Pixel value: grayscale range [0, …, 255]
0 is pure black while 255 is pure white

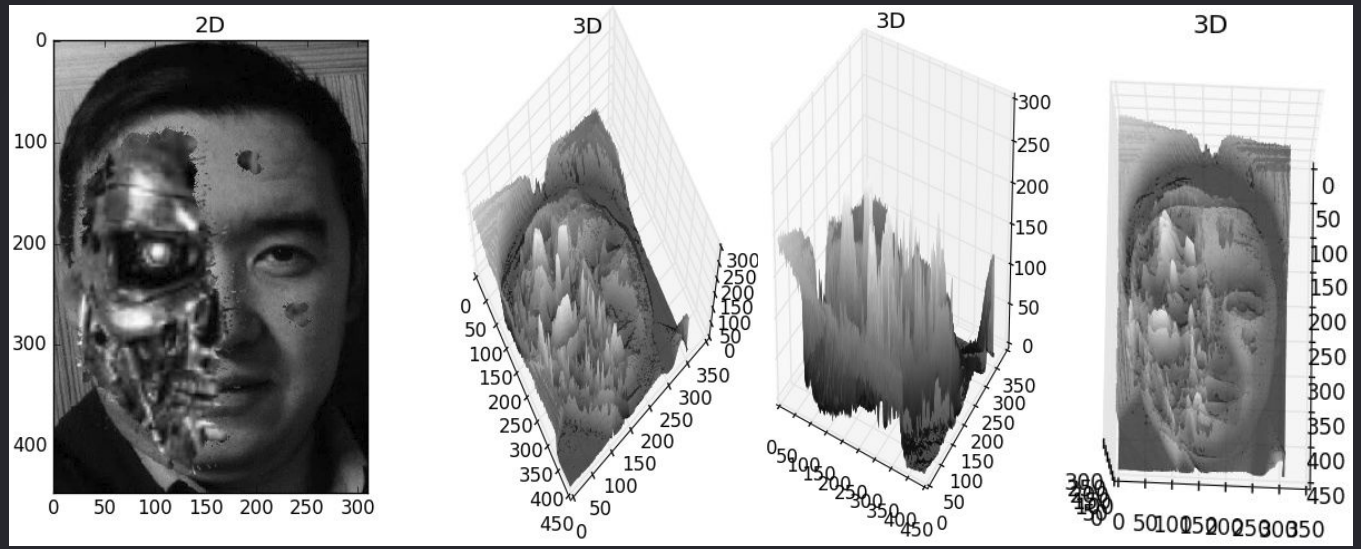

Grayscale Image
Pixel values samples
x=831.571 y=206.583 [237]
x=65.2544 y=385.981 [6]
x=564.689 y=502.257 [107]


Color Image
Each pixel contains 3 numbers [R, G, B]
ranging [0, …, 255]


Color Image
Pixel values samples
x=777.309 y=191.079 [250, 248, 249]
x=95.154 y=387.088 [11, 9, 10]
x=670.999 y=515.546 [93, 89, 86]

An Image as function f from R^2 to R^M:

f(x, y) gives the intensity at position (x, y)



An Image as function f from R^2 to R^M:

f(x, y) gives the intensity at position (x, y)

Grayscale Image has single channel

Color Image has three channels

An Image as function f from R^2 to R^M:

f(x, y) gives the intensity at position (x, y)

Grayscale Image can be represented as 2D Matrix with one channel

Color Image can be represented as 2D Matrix with three channels

An Image as function f from R^2 to R^M:

f(x, y) gives the intensity at position (x, y)

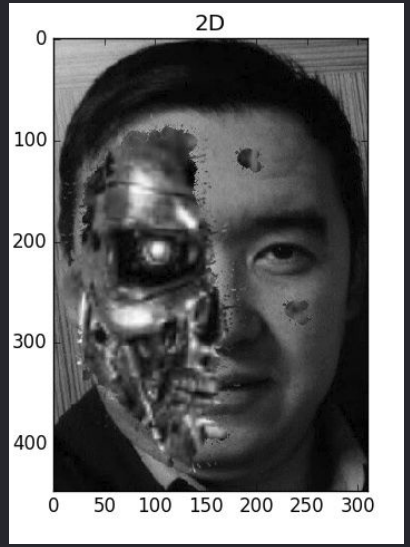
Grayscale Image can be represented as 2D Matrix with one channel

Color Image can be represented as 2D Matrix with three channels

An Image as function f from R^2 to R^M:

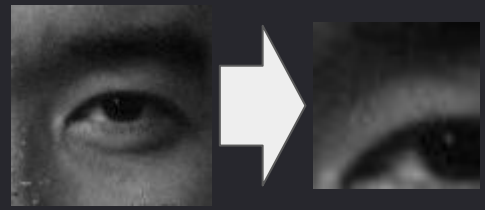
f(x, y) gives the intensity at position (x, y)

(446, 310)



(100, 100)

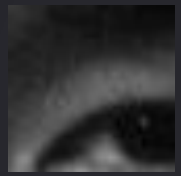
(40, 40)



An Image as function f from R^2 to R^M:

f(x, y) gives the intensity at position (x, y)

(40, 40)



How we see grayscale image

```
@jomart-hp-pc: ~/Desktop
jomart@jomart-hp-pc:~$ cd Desktop
jomart@jomart-hp-pc:~/Desktop$ python crop.py
[[ 56  51  50  48  45  38  32  33  33  26  21  20  20  19  19  19  16  14
   19  20  17  12   4   5  11  11  12  15  14  10   5   3   4   2   1   3
    4   4   2   1]
 [ 55  49  47  45  43  41  37  38  30  20  12  13  16  16  16  17  18  12
   16  17  13  10  10  16  11   8   7   8   9   7   4   3   4   3   2   3
    3   3   2   2]
 [ 48  47  49  47  43  40  36  36  33  24  17  16  15  13  15  19  16  13
   16  13   7  11  16  19  17  11   6   7  10  10   6   3   6   6   6   6
    5   5   5   7]
 [ 41  45  50  46  40  35  31  32  30  24  18  14  10   7   9  13  10   9
   14  12   5   8  14  12  14  11   8   8   8   7   6   5   6   6   6   6
    5   4   5   7]
 [ 41  43  44  38  32  30  29  31  29  26  21  18  17  17  16  14  15   7
   11  18  11   7  11  13   9  11  12  10   5   3   6   9   5   5   6   6
    7   6   6   7]
 [ 43  41  39  33  29  29  26  26  23  24  23  20  19  20  19  17  21  13
   11  16  13  10  16  18  15  14  12  10   8   7   8   9   8   8   8  10
   11  11  11  11]
 [ 40  39  39  36  32  28  23  23  23  26  26  23  21  21  21  21  20  17
   12  11   9  11  18  17  18  15  12  11  12  13  12   9  11  11  11  13
   14  15  16  17]
 [ 36  37  40  38  32  28  25  27  28  26  24  24  24  24  23  23  24  21
   14  15  15  12  18  19  15  17  18  18  18  19  20  20  16  16  17  19
   20  21  23  25]
 [ 35  36  37  33  26  27  30  29  31  27  27  22  25  24  22  25  27  19
   22  27  22  18  22  26  22  18  24  19  19  22  26  25  21  19  22  24
   25  28  28  30]
 [ 30  33  29  24  19  22  29  31  31  28  28  21  24  25  25  28  29  22
   25  30  26  22  26  29  25  21  26  20  20  22  28  30  25  23  27  28
   28  30  31  32]
 [ 34  27  23  23  24  27  30  30  27  27  30  25  28  29  28  29  33  27
   28  31  27  25  28  31  27  25  31  29  27  28  32  35  34  32  36  38
   37  37  37  38]
 [ 25  23  26  31  33  31  29  29  27  27  30  28  31  30  31  33  33  30
   31  32  29  29  34  37  35  34  37  38  36  34  35  37  42  41  46  49
   49  49  49  50]
 [ 25  26  29  32  31  27  28  33  33  30  28  28  31  28  34  40  30  32
   36  38  38  40  44  47  49  48  45  46  44  42  42  43  45  44  52  57
   58  59  59  62]
 [ 31  29  27  28  31  30  31  36  36  32  30  32  35  30  36  43  32  38
   44  48  50  53  54  54  59  62  58  61  60  58  56  56  53  51  56  60
```

First line - grayscale values

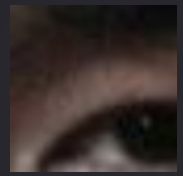
Second line - grayscale values

How computer see

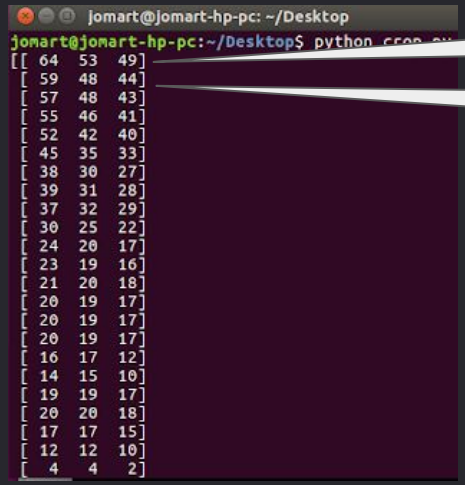
An Image as function f from R^2 to R^M:

f(x, y) gives the 3 values at position (x, y)

(40, 40)



How we see color image



First line - First pixel values

First line - Second pixel values

How computer see

An Image as function f from R^2 to R^M:

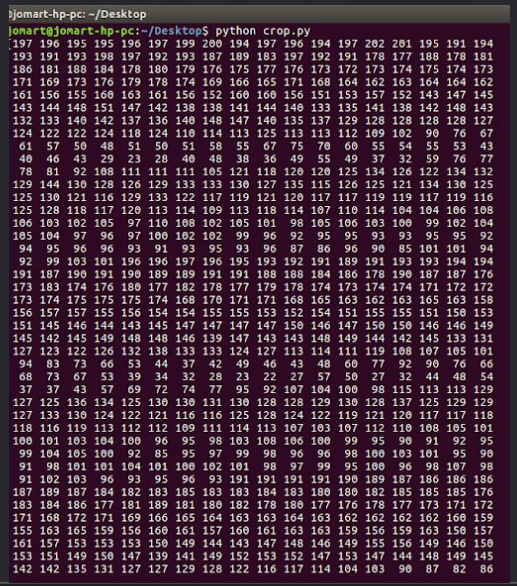
We can convert values in each location

An Image as function f from R^2 to R^M:

We can convert values in each location

An Image as function f from R^2 to R^M:

Image thus represented as matrix of integer values



Noise in images:

Noise is just another function that is combined with the original to get a new function

$$I'^{\rightarrow}(x,y) = I^{\rightarrow}(x, y) + {}^{\rightarrow}(x, y)$$

New Image | Image | Noise

Some types of Noises:

Salt & Pepper noise: random occurence of black and white pixels

Some types of Noises:

Impulse noise: random occurence of white pixels

Some types of Noises:

Natural (snow), not Impulse noise

Some types of Noises:

Gaussian noise: random variations in intensity drawn from a Gaussian normal distribution

Agenda

Week #2 - Computer Vision - Lecture
Image Processing for Computer Vision

1. Linear Image processing: Images as function
2. Linear Image processing: Filtering

Form a new image whose pixels are a combination original pixel values

Goal: modify or enhance image properties

2D discrete-space systems (filters)

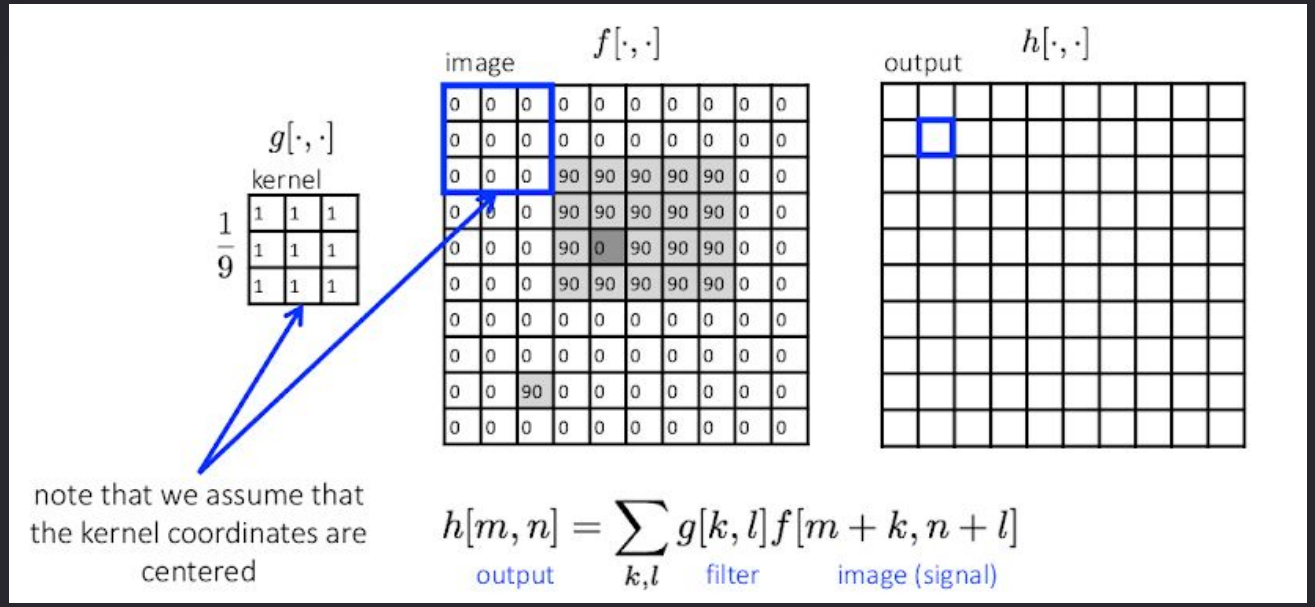
Sample

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

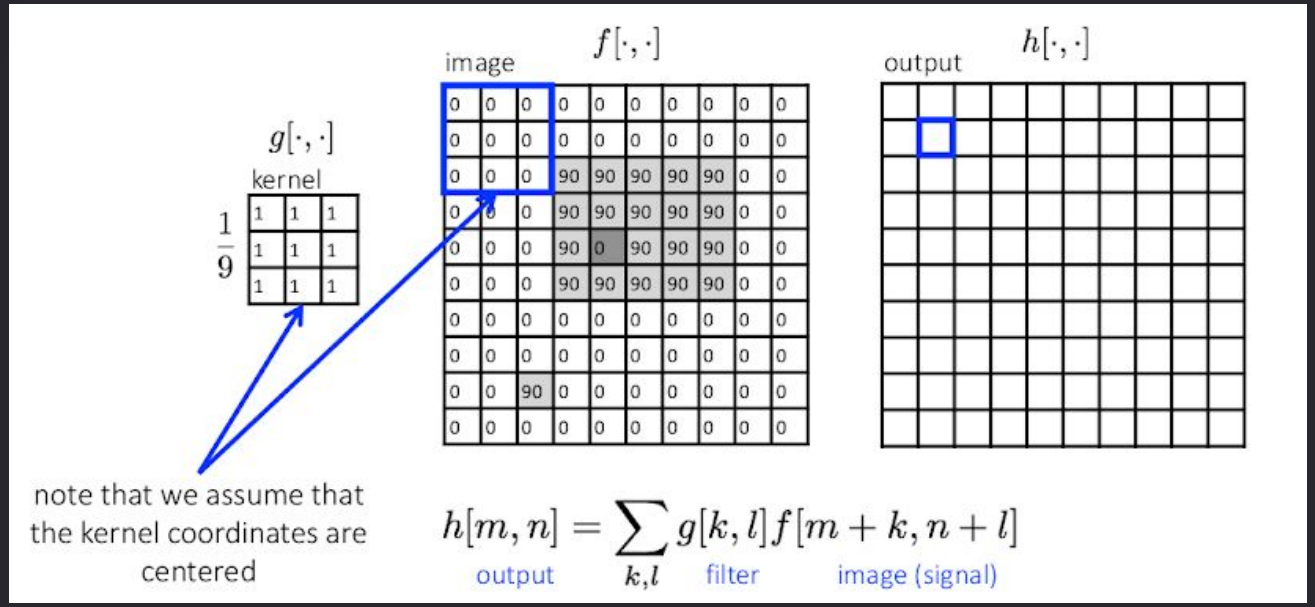
$$g = \mathcal{S}[f], \quad g[n,m] = \mathcal{S}\{f[n,m]\}$$

$$f[n,m] \xrightarrow{\mathcal{S}} g[n,m]$$

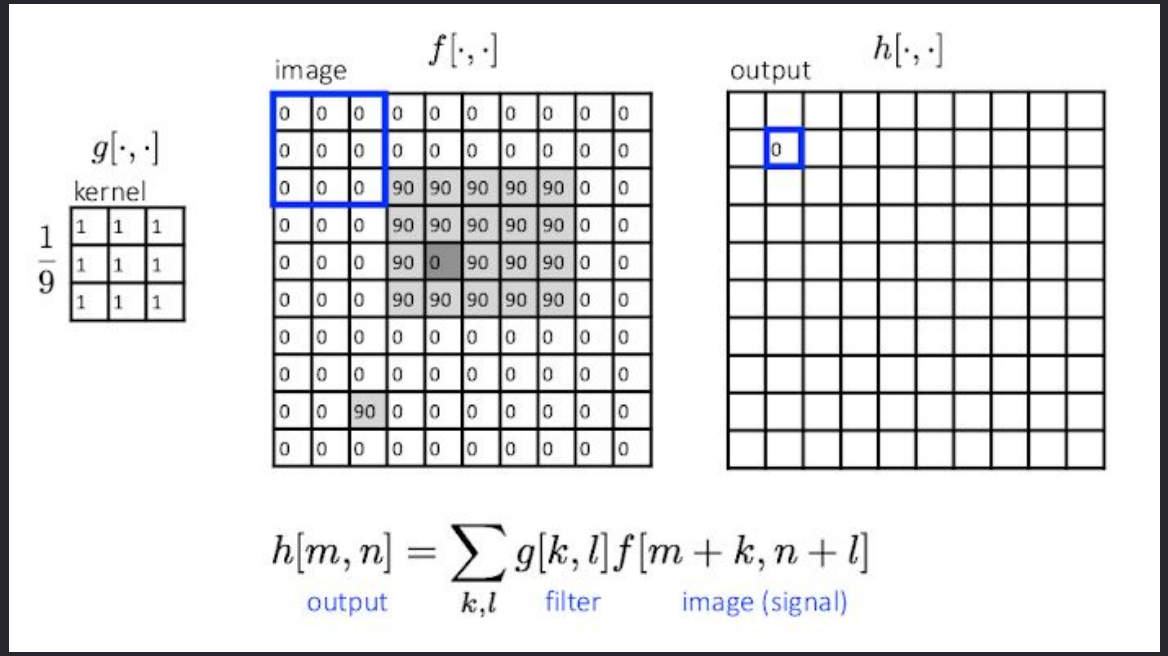
Sample: Box filter - Moving Average in 2D



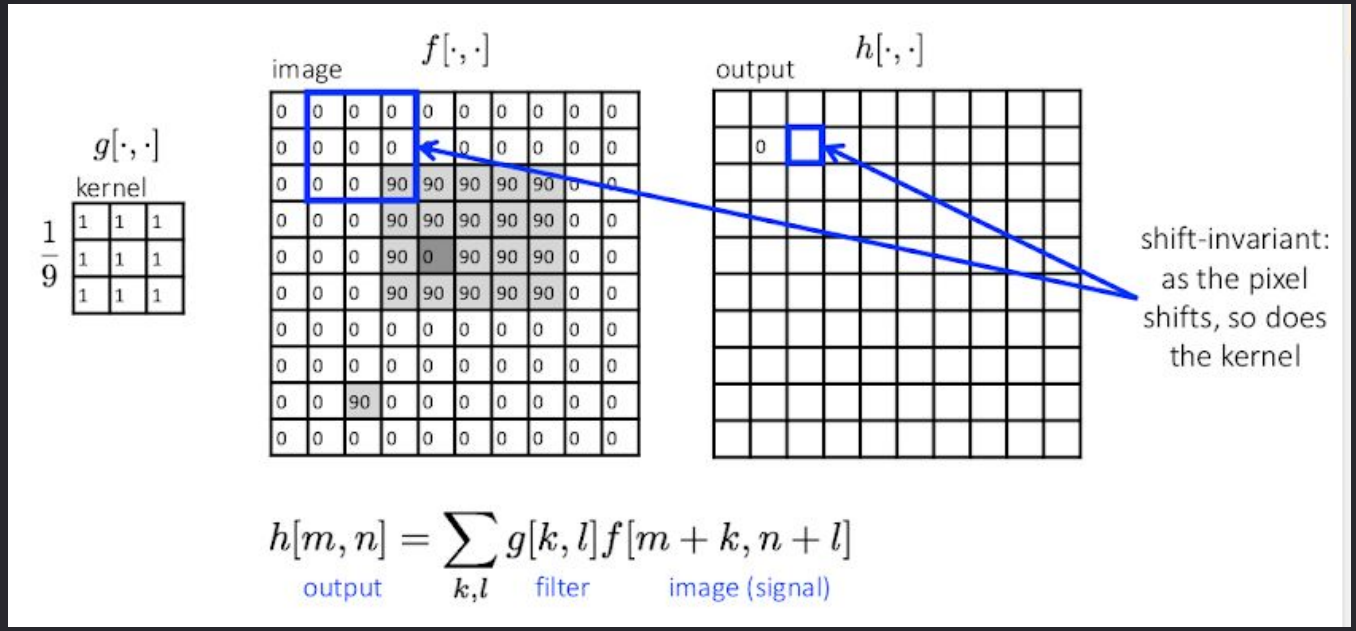
Sample: Box filter - Moving Average in 2D



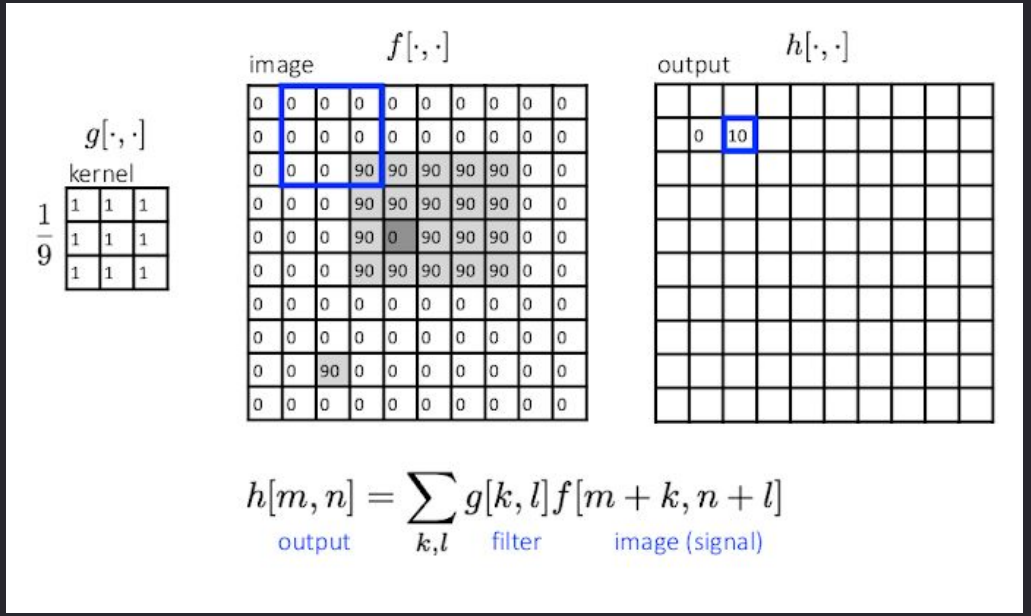
Sample: Box filter - Moving Average in 2D



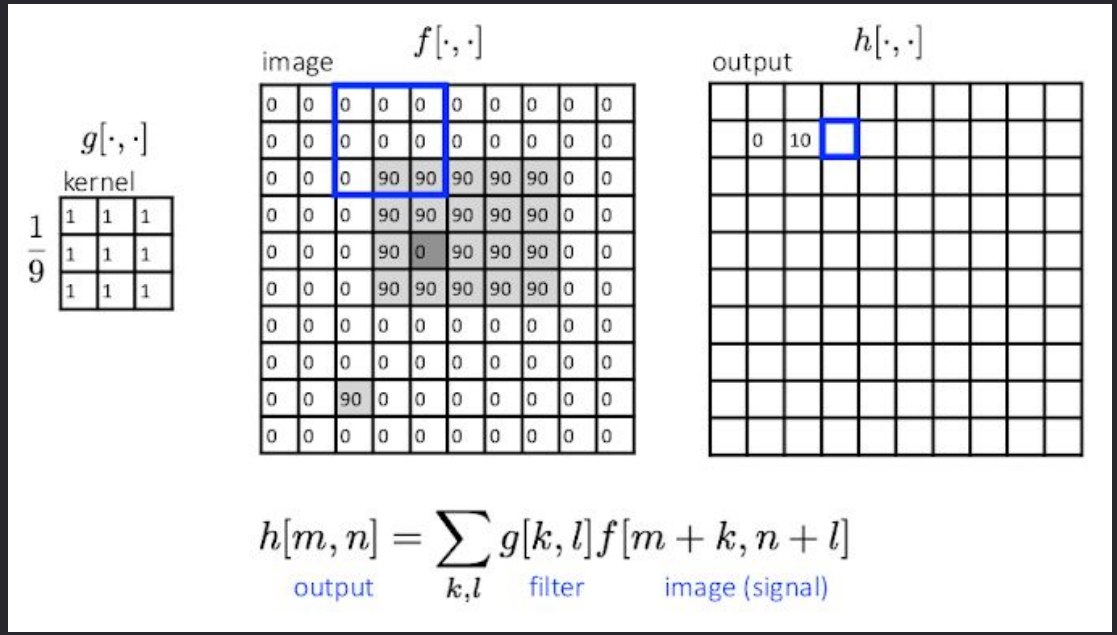
Sample: Box filter - Moving Average in 2D



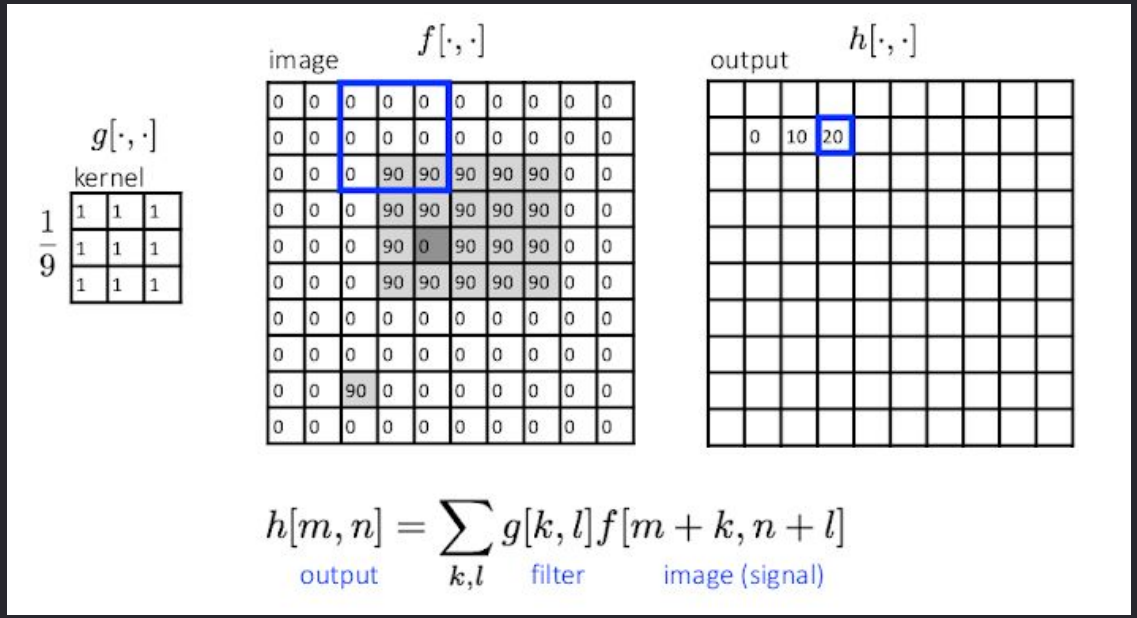
Sample: Box filter - Moving Average in 2D



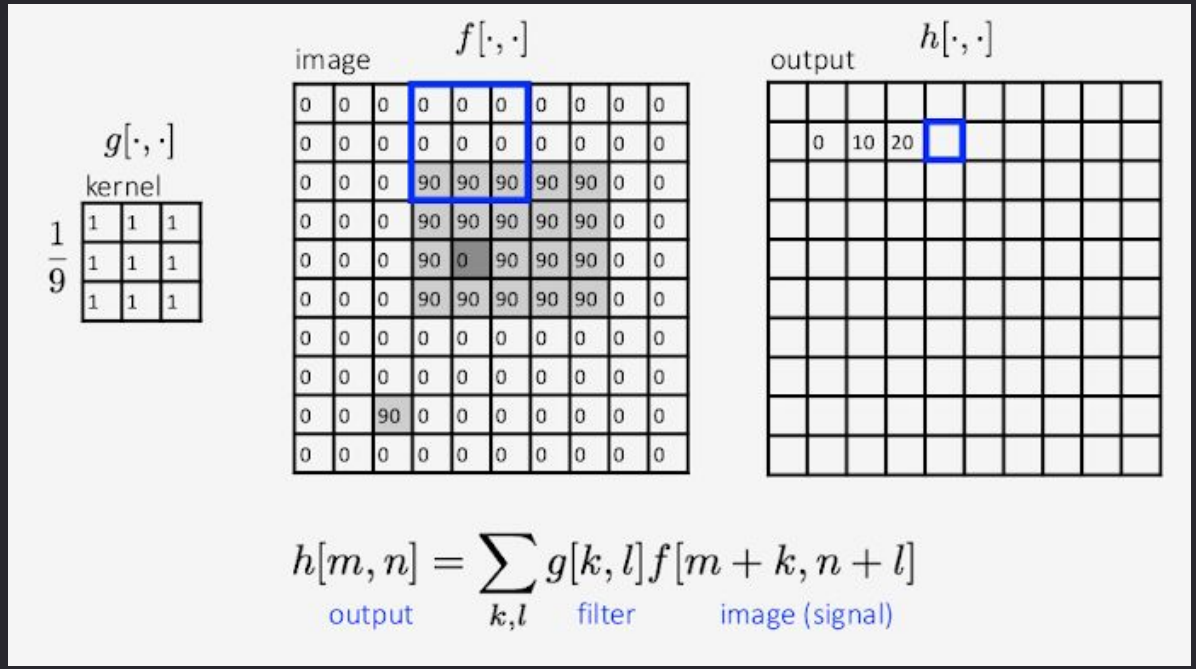
Sample: Box filter - Moving Average in 2D



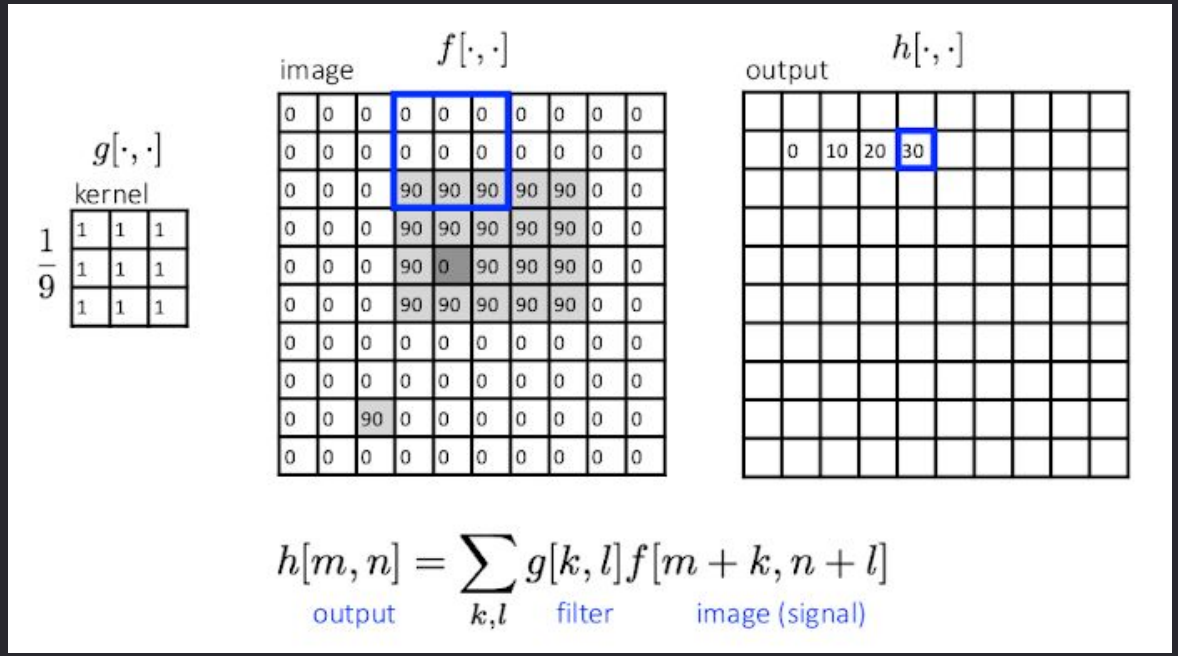
Sample: Box filter - Moving Average in 2D



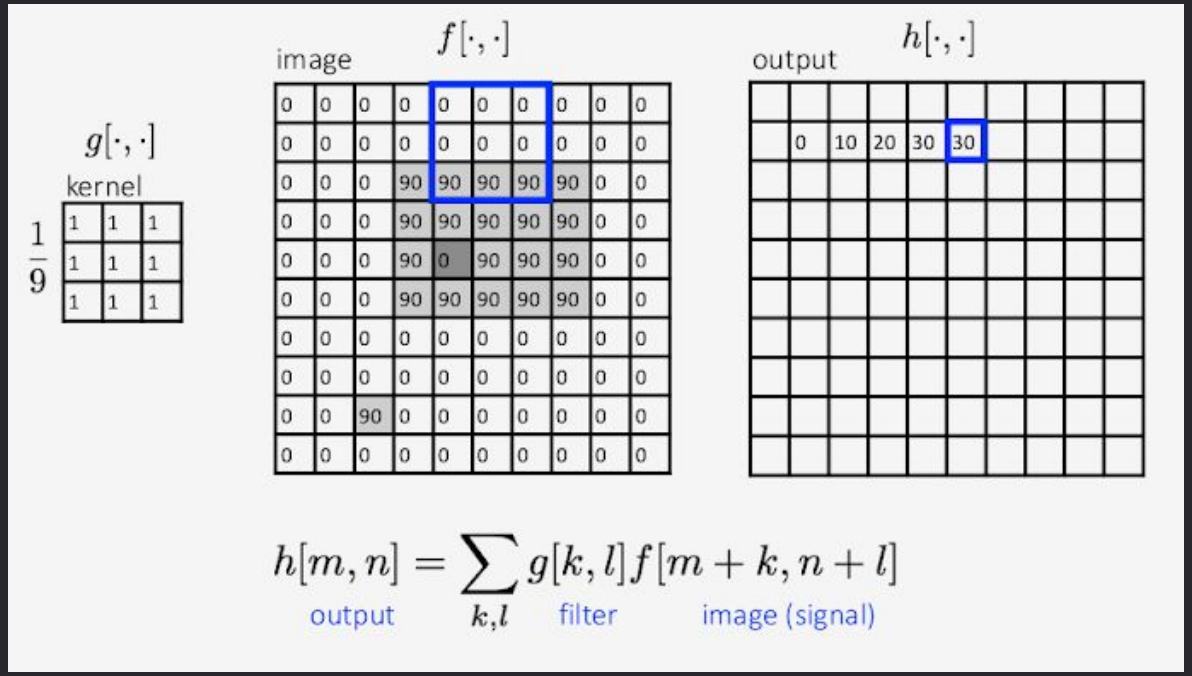
Sample: Box filter - Moving Average in 2D



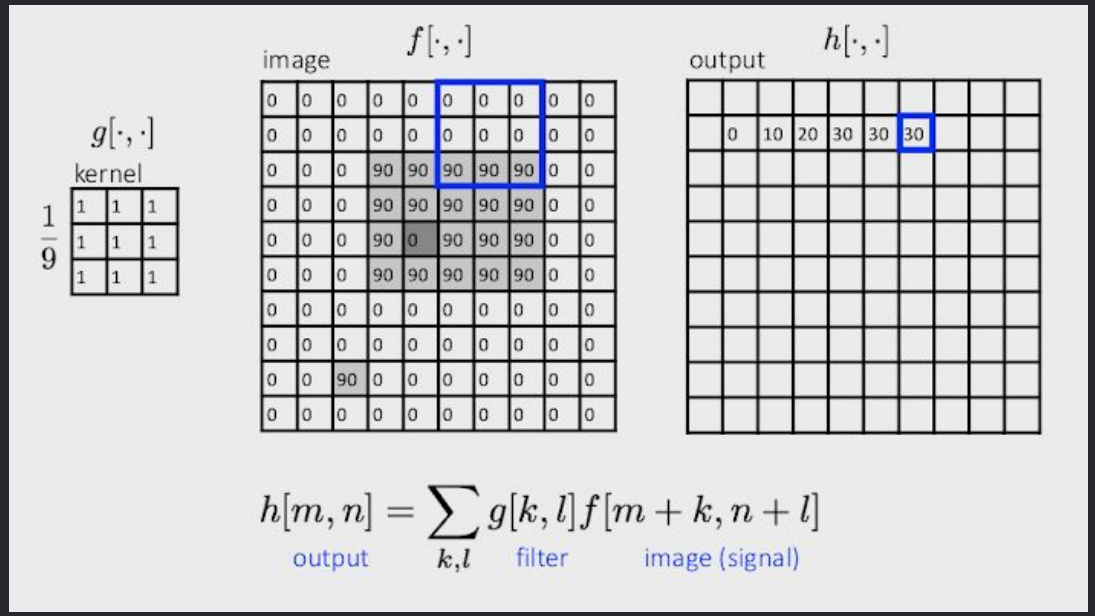
Sample: Box filter - Moving Average in 2D



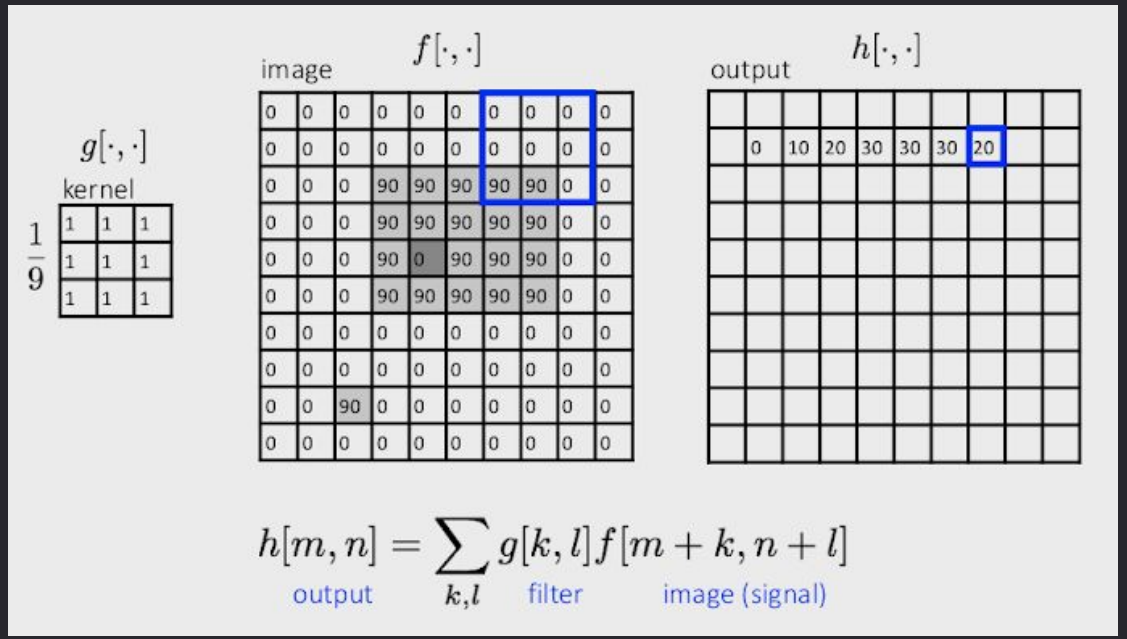
Sample: Box filter - Moving Average in 2D



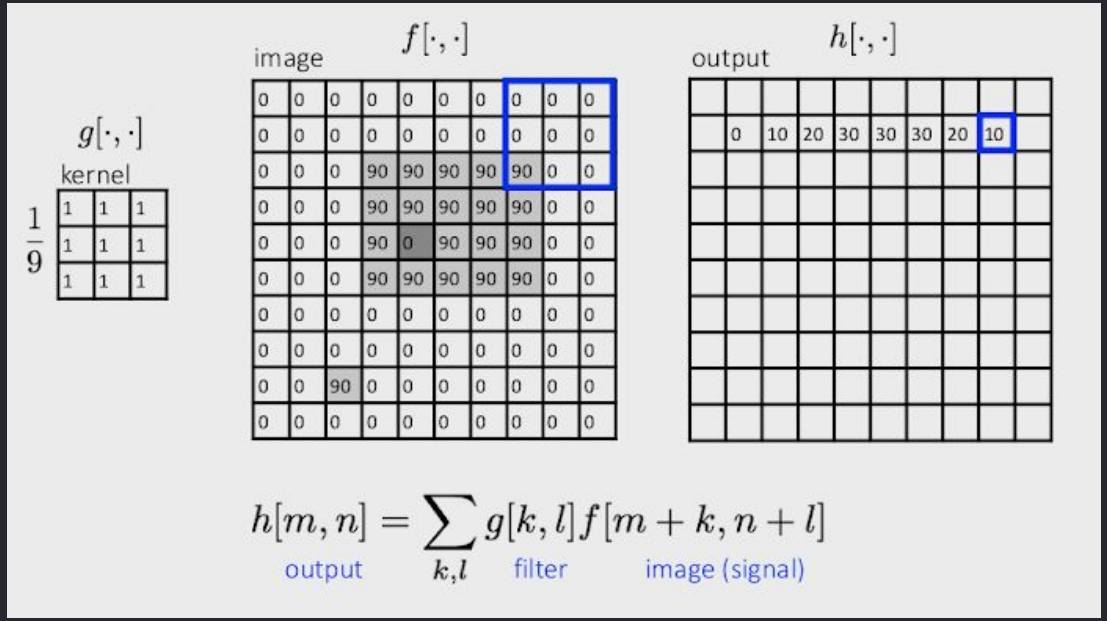
Sample: Box filter - Moving Average in 2D



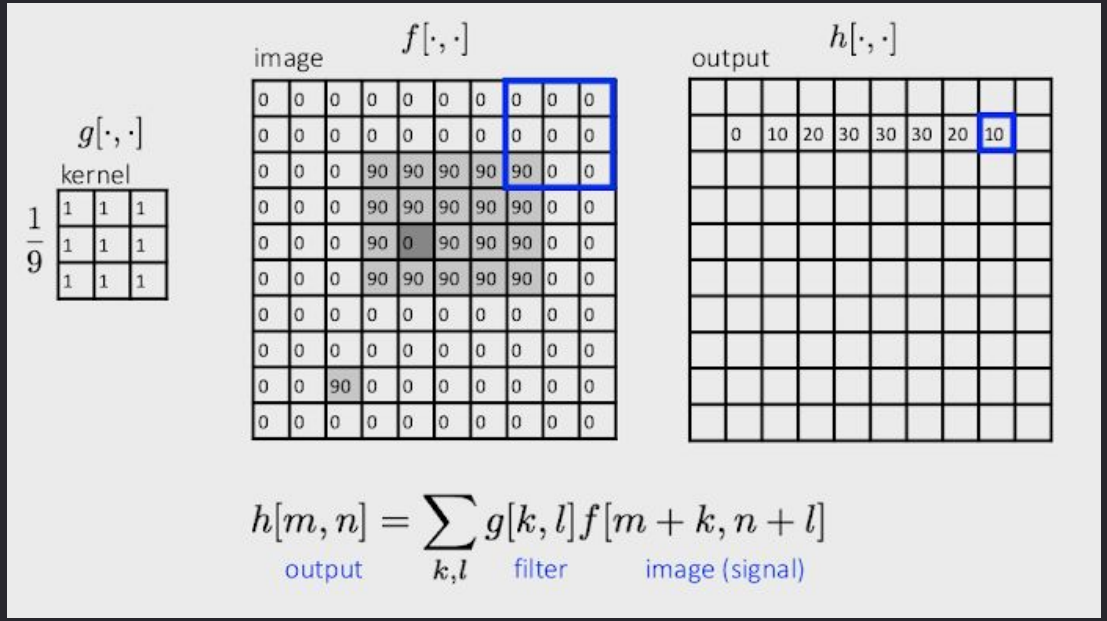
Sample: Box filter - Moving Average in 2D



Sample: Box filter - Moving Average in 2D



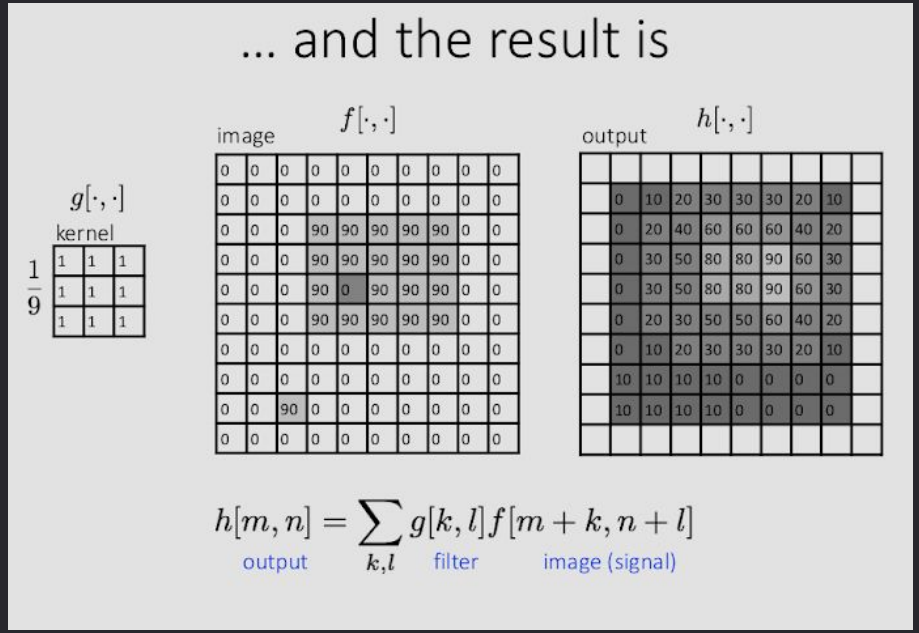
Sample: Box filter - Moving Average in 2D



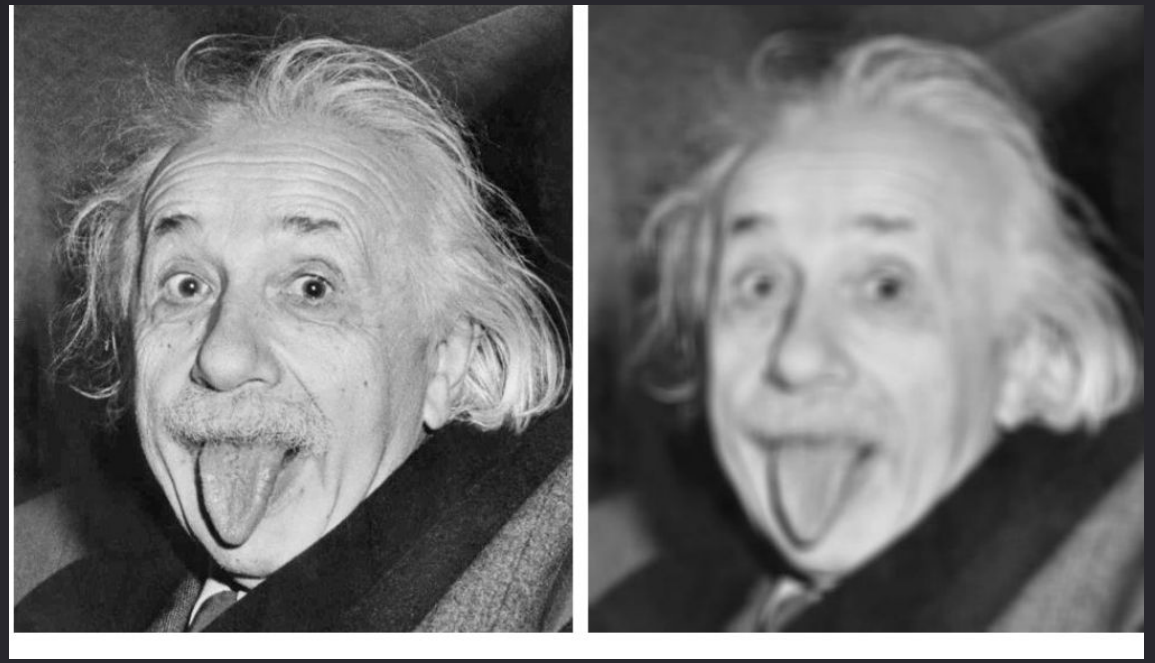
Sample: Box filter - Moving Average in 2D

continuing for each line ...

Sample: Box filter - Moving Average in 2D



Sample: Applying Box filter result



Sample: Applying Box filter result

Sample: Applying Box filter result

Sample: Applying Box filter result

Sample: Image segmentation

$$g[n,m] = \begin{cases} 255, & f[n,m] > 100 \\ 0, & \text{otherwise.} \end{cases}$$

Sample: Image segmentation

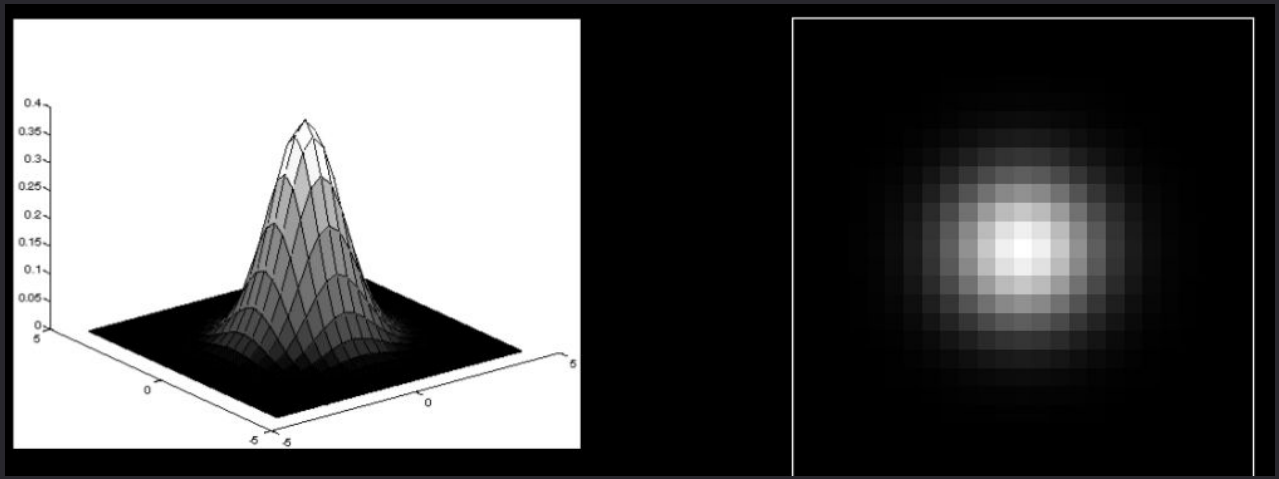
Sample: Gaussian filter

nearest neighboring pixels have the most influence

Sample: Gaussian filter

an Isotropic Gaussian

Sample: Gaussian filter



Sample: Gaussian filter

Sample: Gaussian filter

Resources & References |

1. https://www.forbes.com/sites/markhughes/2016/01/28/how-to-save-the-terminator-franchise/#692400624f72
2. https://www.udacity.com/course/introduction-to-computer-vision--ud810

Thank you for your time & attention!