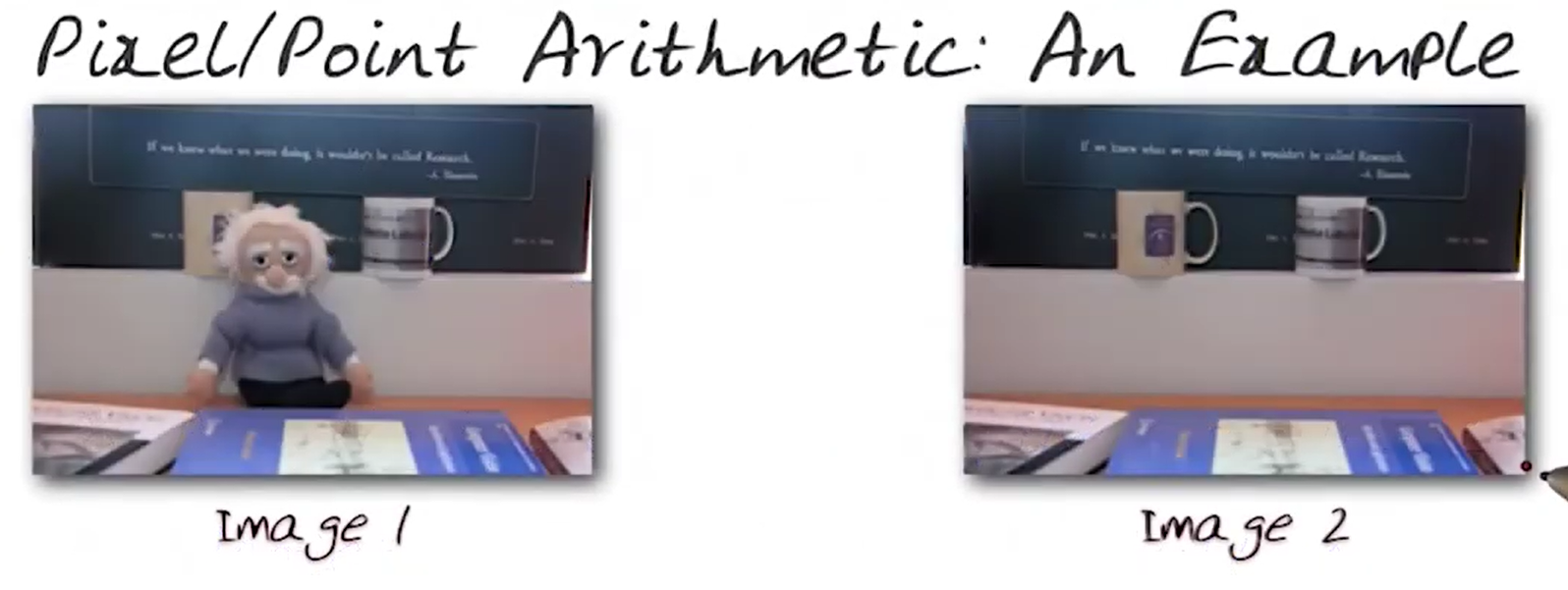
**Computation Photography**

***Blending Modes***

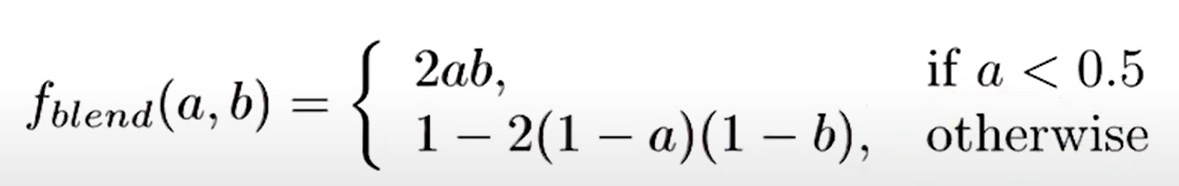
Variety of blending modes built on concept of point processes



1. Images manipulated in blending techniques will have the same size, and same pixel value. Except one image has some extra information like above image
2. Blend two pixels from two images Fblend(a,b)
3. Average blending Fblend(a,b) = (a + b) / 2
4. Normal blending Fblend(a,b) = b

**Arithmetic blend modes**

1. Divide (brighten photos)
2. Addition (too many whites in the photos)
3. Subtract (too many blacks in the photos)
4. Difference (subtract with scaling), when the value become zero we scale the value to some degree to avoid blacks in the photos
5. Darken Fblend (a,b) = min(a, b) for RGB
6. Lighten Fblend (a,b) = max(a, b) for RGB
7. Multiply, it generates darker image Fblend = ab
8. Screen, it generates brighter image Fblend = 1 – (1-a)(1-b), considering the value 0 to 1
9. Overlay, The parts of the top layer where the base layer is light become lighter where the base layer is dark become darker, below is equation for overlay method which has the combination of both multiply and screen method



**Dodge and Burn**

1. Dodge and burn change the lightness of the pictures
2. Dodge lightens an image, while burning darkens the image
3. Dodge builds on screen mode
4. Burn builds on multiply mode
5. There are numerous variations of each.

***Smoothing***

**Digital image is a function**

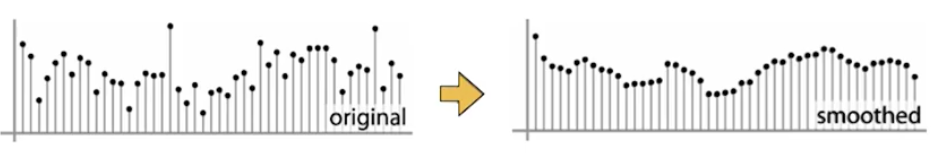
1. Smoothening of an image is achieved with the help of neighbouring pixels

**Smoothing Types**

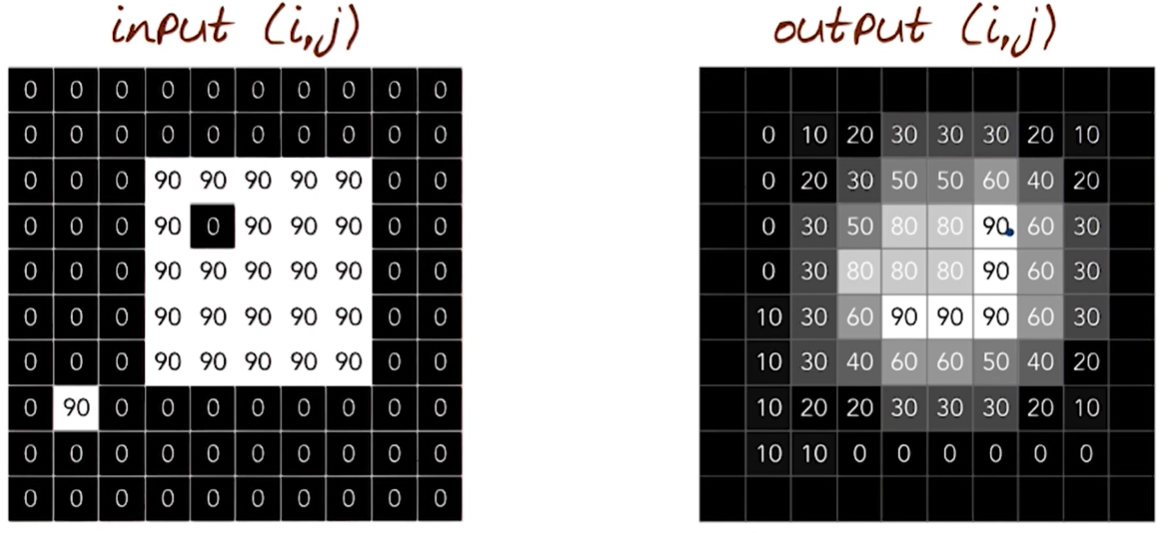
1. Moving average [ 1, 1, 1, 1, 1] / 5 Since we are giving equal weight to all pixel we will get a flat image
2. Weighted moving average [ 1, 4, 6, 4, 1] \* 1 / 16 , value 16 comes with the sum of [ 1, 4, 6, 4, 1], here we are giving a ramp for the pixel that is subjected for smooting
3. Above shown values are suitable for 1d array, we should be having 2 \* 2 array to apply on an image



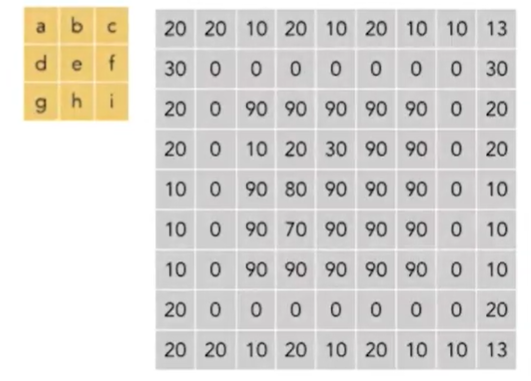
1. Below image shows the original and smoothed signal of an image



**Smoothing Process over an image using averages**



1. While applying kernel to an image, we need to pad some values to edges of an image, we can use following methods for padding the edges Wrap around, Copy edges, reflect across (refer and update)
2. The array or image that we rub over an image is called kernel, the yellow array below is called kernel

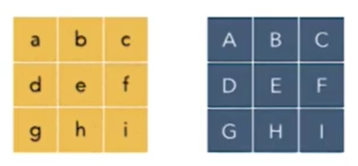
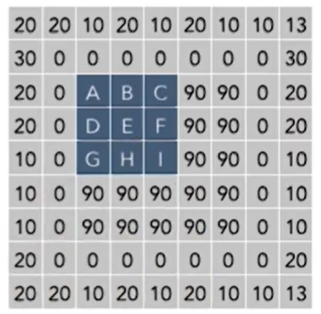


1. The kernel is represented as function h (i, j)
2. For 3 \* 3 matrix the neighbour around the original pixel is 1 and it is represented as variable K
3. For 3 \* 3 kernel K value is 1,and for 5 \* 5 is 2 and so on
4. Window size is 2K + 1, therefore for K = 1 our window size is 3 \* 3, Window is referring our kernel

**Below are the function referred for input, output and kernel image**

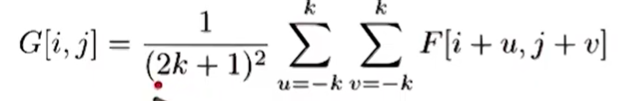
1. Input image F [i, j]
2. Output image G [i, j]
3. Kernel h [i, j]

**The mathematical representation of kernel and the pixel being operated on**



Let’s take the pixel E from above image and apply the kernel [a, b, c, d, e, f, g, h, i], we get below equation

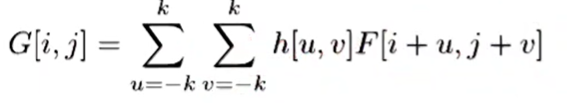
E = ( ( a \* A) + (b \* B) + ( c \* C) + (d \* D) + ( e \* E) + (f \* F) + ( g \* G) + (h \* H) + ( i \* I) ) / 9



We loop over all the pixels in neighbourhood around image pixel F [i, j]

1 /( (2K + 1) pow 2) is attribute uniform weight to each pixel

**Below is the more generalized form of the equation**



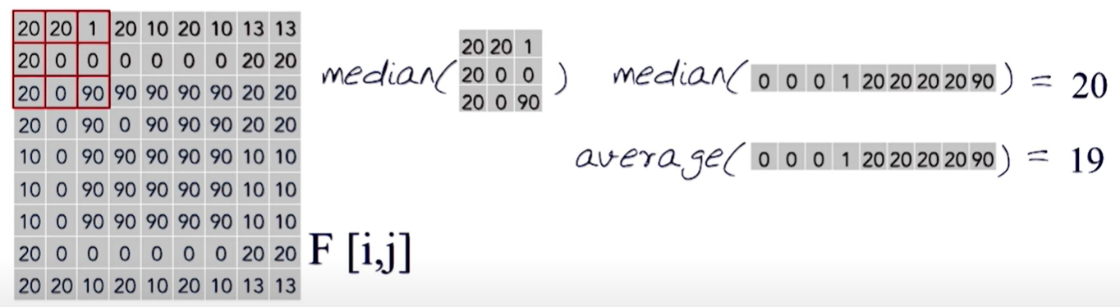
h [u, v] is attribute non-uniform weight, it is referred to as cross-correlation

**Median Filter**

Median filtering is achieved by taking all the neighbouring pixel of the pixel that we want to normalize and sort the all values (we will be getting 10 values if take 3 \* 3 ) and take the mid values as the normalized values

Some advantages of median filter

1. Reduces noise
2. Preserves edges (sharp lines)



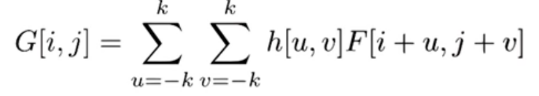
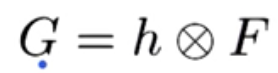
Compared to average filter, median filter helps in reducing the noise and sharpens the edge

***Convolution and Cross-Correlation***

**Cross-Correlation**

In signal processing, cross-correlation is a measure of similarity of two waveforms as a function of a time-lag applied to one of them, which means combining two different waveforms to figure out the best ways to correlate two different signal together and allowing us to measure the similarity between those signals.

Also knows as sliding dot product or sliding inner-product of two different signal

 Denoted by 

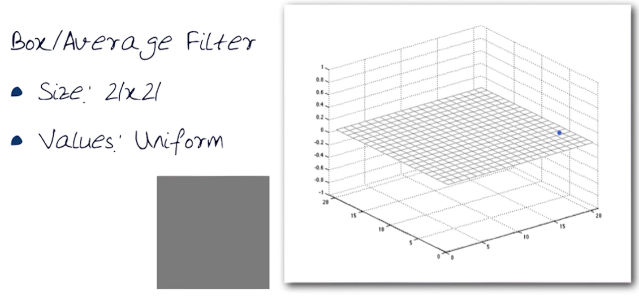
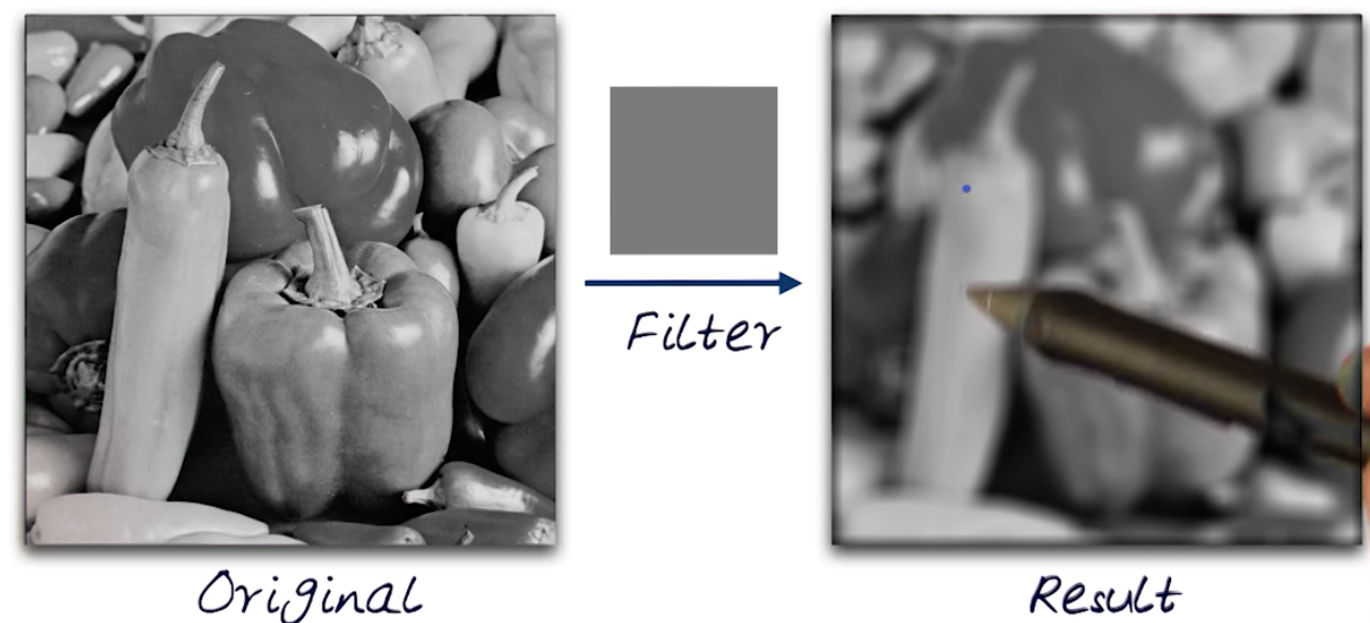
Filtering an image

1. Replace each pixel with a linear combination of its neighbour

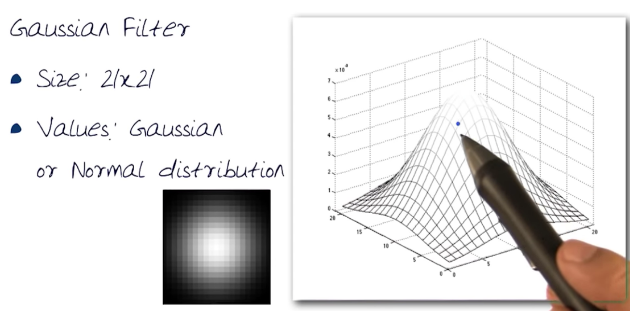
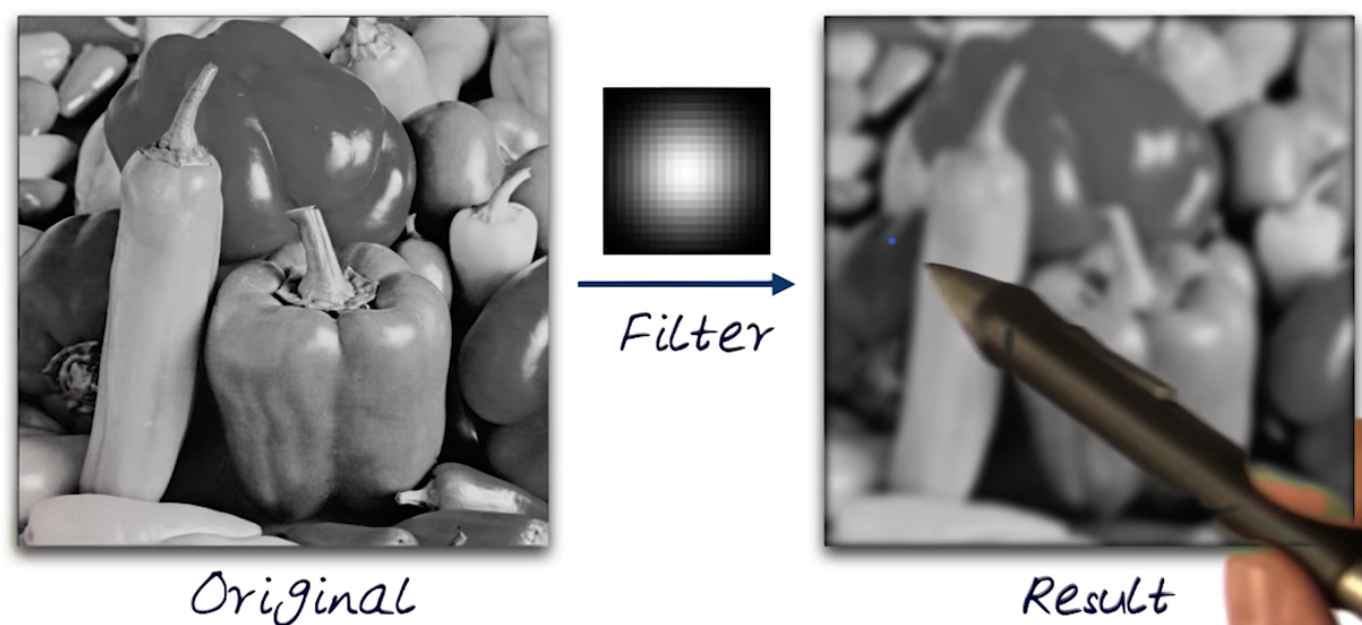
Filter “Kernel” or “Mask”

1. h [u, v] is the prescription for weights in the linear combination

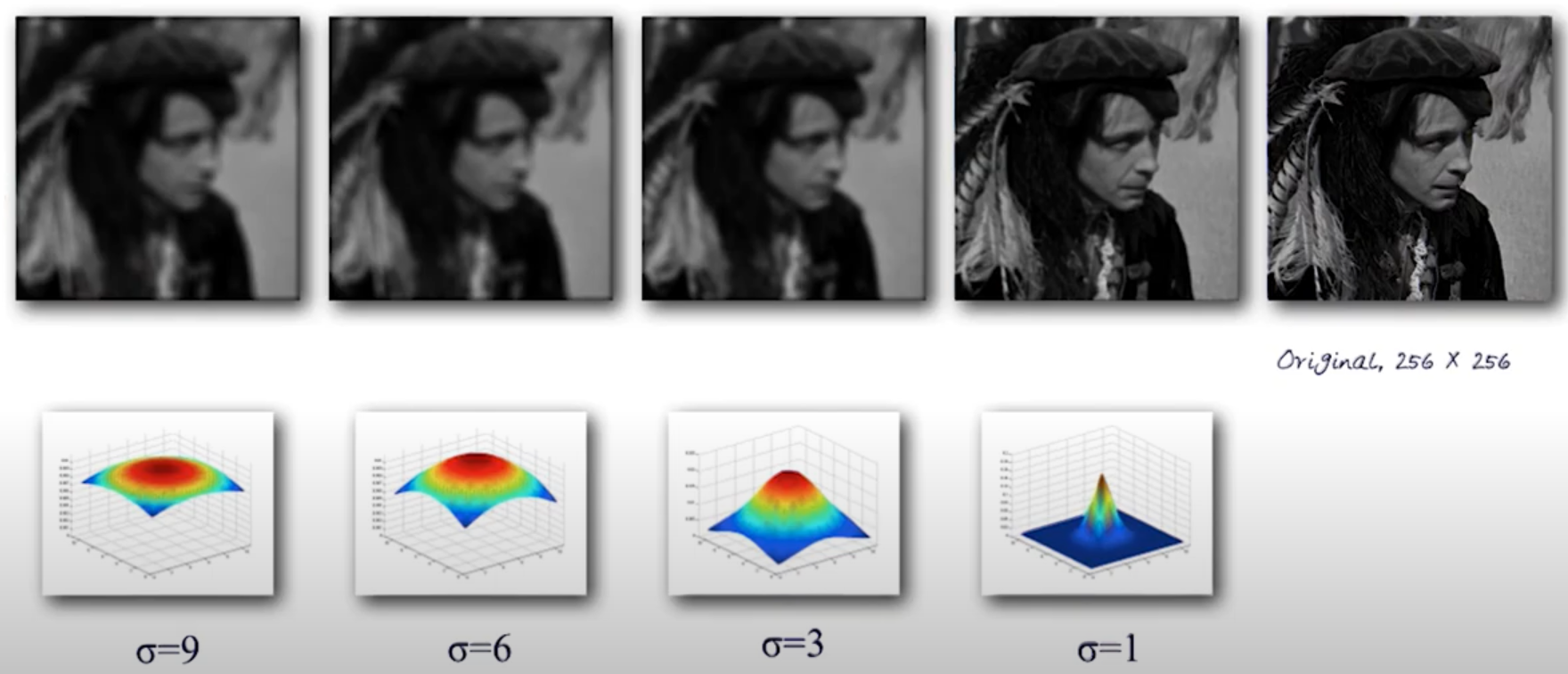
Box Filter



Gaussian Filter



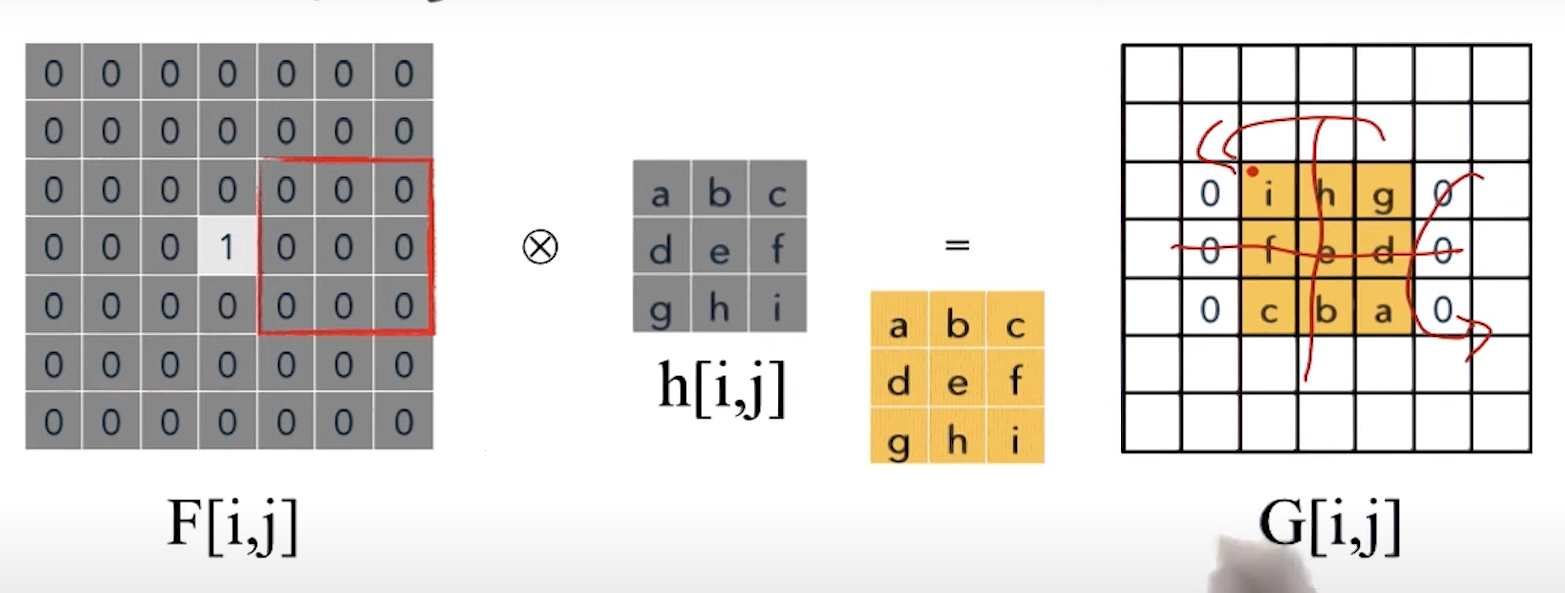
Gaussian Filter for Smoothing



Sigma determines extent of smoothing

Filtering By Kernel (Defining Convolution)

The below image with all 0 with 1 in the middle is referenced as impulse image

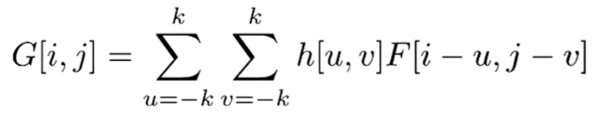
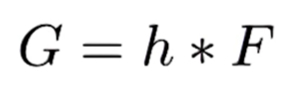


Filter means to slide the kernel over the image, Results in a reversed response

Convolution method

1. Convolution is a mathematical operation on two functions F and h
2. Produces a third function that is typically viewed as a modified version
3. Gives the area of overlap between the two function
4. In a form of the amount that one of the original functions is translated

Mathematical Notation

Denoted by   
In convolution method, the kernel operation is as follows

1. Flip filters in both dimensions
2. Bottom to top
3. Right to left
4. Then apply cross correlation

Third box is the final kernel state at the end of first step, and the cross correlation is calculated with the final kernel



Convolution vs Cross Correlation

