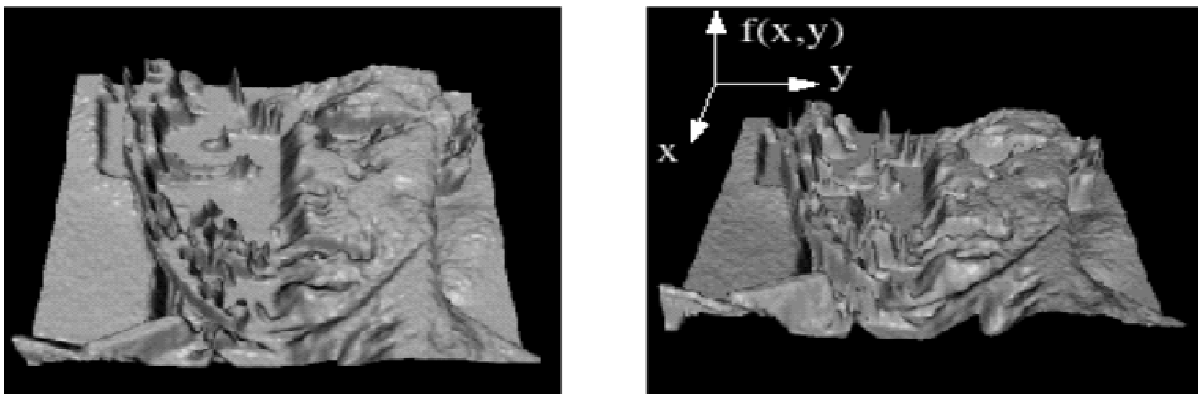
# Linear Filters and Non linear Filters

## Images as functions

- Images are a function $f$, from $R^2 \to R : f(x, y)$, that gives the intensity at a position $(x, y)$.
- Realistically, we expect the image only to be defined over a rectangle, with a finite range: $f; [a, b] \times [c, d] \to [0, 1]$, here $x, y$ from $f(x, y)$ are defined over the ranges $[a, b]$ and $[c, d]$ respectively.
- A color image is just three functions pasted together. We can write this as a “vector-valued” function:

$$f(x, y) = \begin{bmatrix} r(x, y) \\ g(x, y) \\ b(x, y) \end{bmatrix}$$

- 
- Like for example:



-

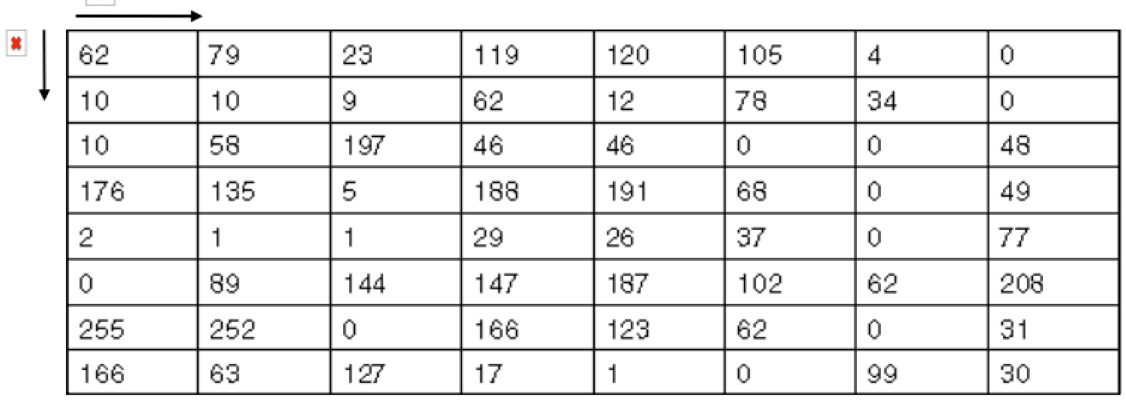
## Digital Images:

- Digital images are discretization of the intensities.

- We usually work with **digital** (**discrete**) images:
  - **Sample** the 2D space on a regular grid
  - **Quantize** each sample (round to nearest integer)
- If our samples are $\Delta$ apart, we can write this as:

$$f[i,j] = \text{Quantize}\{f(i\Delta, j\Delta)\}$$

- The image can now be represented as a matrix of integer values



| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 62 | 79 | 23 | 119 | 120 | 105 | 4 | 0 |
| 10 | 10 | 9 | 62 | 12 | 78 | 34 | 0 |
| 10 | 58 | 197 | 46 | 46 | 0 | 0 | 48 |
| 176 | 135 | 5 | 188 | 191 | 68 | 0 | 49 |
| 2 | 1 | 1 | 29 | 26 | 37 | 0 | 77 |
| 0 | 89 | 144 | 147 | 187 | 102 | 62 | 208 |
| 255 | 252 | 0 | 166 | 123 | 62 | 0 | 31 |
| 166 | 63 | 127 | 17 | 1 | 0 | 99 | 30 |

## Mean Filtering:

- $F[x,y] \rightarrow G[x,y]$, This is the transformation.

$F[x,y]$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 0 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 90 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$G[x,y]$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| | 0 | 10 | 20 | 30 | 30 | 30 | 20 | 10 | |
| | 0 | 20 | 40 | 60 | 60 | 60 | 40 | 20 | |
| | 0 | 30 | 60 | 90 | 90 | 90 | 60 | 30 | |
| | 0 | 30 | 50 | 80 | 80 | 90 | 60 | 30 | |
| | 0 | 30 | 50 | 80 | 80 | 90 | 60 | 30 | |
| | 0 | 20 | 30 | 50 | 50 | 60 | 40 | 20 | |
| | 10 | 20 | 30 | 30 | 30 | 30 | 20 | 10 | |
| | 10 | 10 | 10 | 0 | 0 | 0 | 0 | 0 | |
| | | | | | | | | | |

## Cross-Correlation filtering

https://www.math3ma.com/blog/the-tensor-product-demystified

- As an equation: Assume the window is (2k+1)x(2k+1):

$$G[i,j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$$

- We can generalize this idea by allowing different weights for different neighboring pixels:

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v] F[i+u, j+v]$$

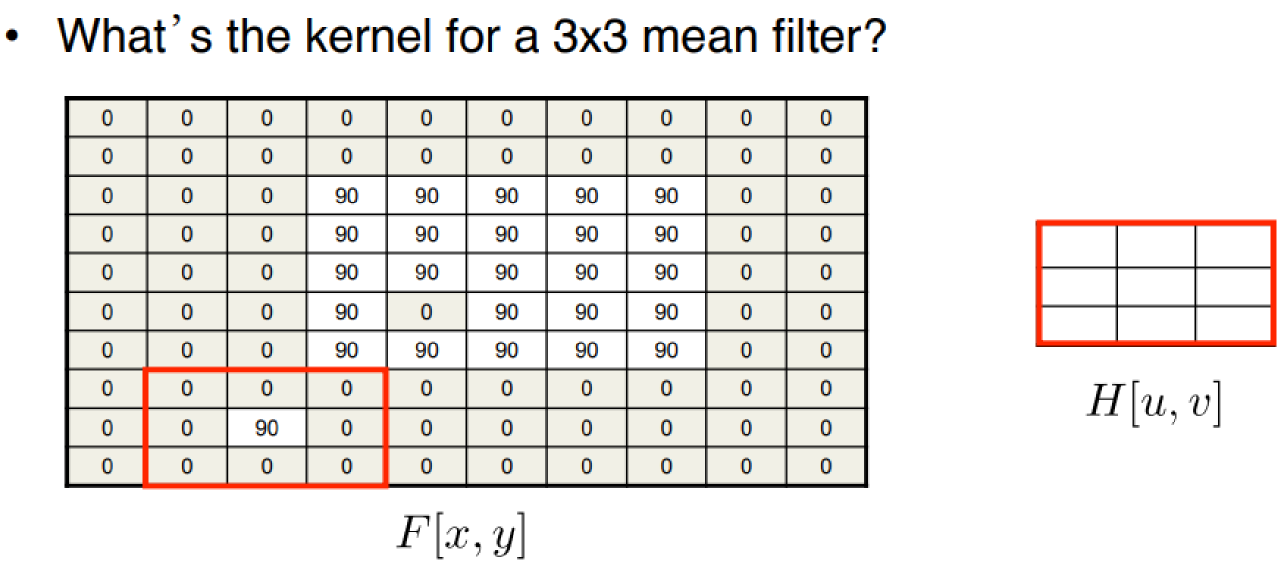
- This is called a **cross-correlation** operation and written:

$$G = H \otimes F$$

- H is called the **filter, kernel,** or **mask**.

- $\otimes$ represents the tensor product between two vectors (IN matrices it is kroneckers product).
- Example:

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} \otimes \begin{bmatrix}0 & 5\\6 & 7\end{bmatrix} = \begin{bmatrix} 1\begin{bmatrix}0 & 5\\6 & 7\end{bmatrix} & 2\begin{bmatrix}0 & 5\\6 & 7\end{bmatrix} \\ 3\begin{bmatrix}0 & 5\\6 & 7\end{bmatrix} & 4\begin{bmatrix}0 & 5\\6 & 7\end{bmatrix}\end{bmatrix} = \begin{bmatrix}1\times 0 & 1\times 5 & 2\times 0 & 2\times 5\\ 1\times 6 & 1\times 7 & 2\times 6 & 2\times 7\\ 3\times 0 & 3\times 5 & 4\times 0 & 4\times 5\\ 3\times 6 & 3\times 7 & 4\times 6 & 4\times 7\end{bmatrix} = \begin{bmatrix}0 & 5 & 0 & 10\\ 6 & 7 & 12 & 14\\ 0 & 15 & 0 & 20\\ 18 & 21 & 24 & 28\end{bmatrix}.$$



- What's the kernel for a 3x3 mean filter?

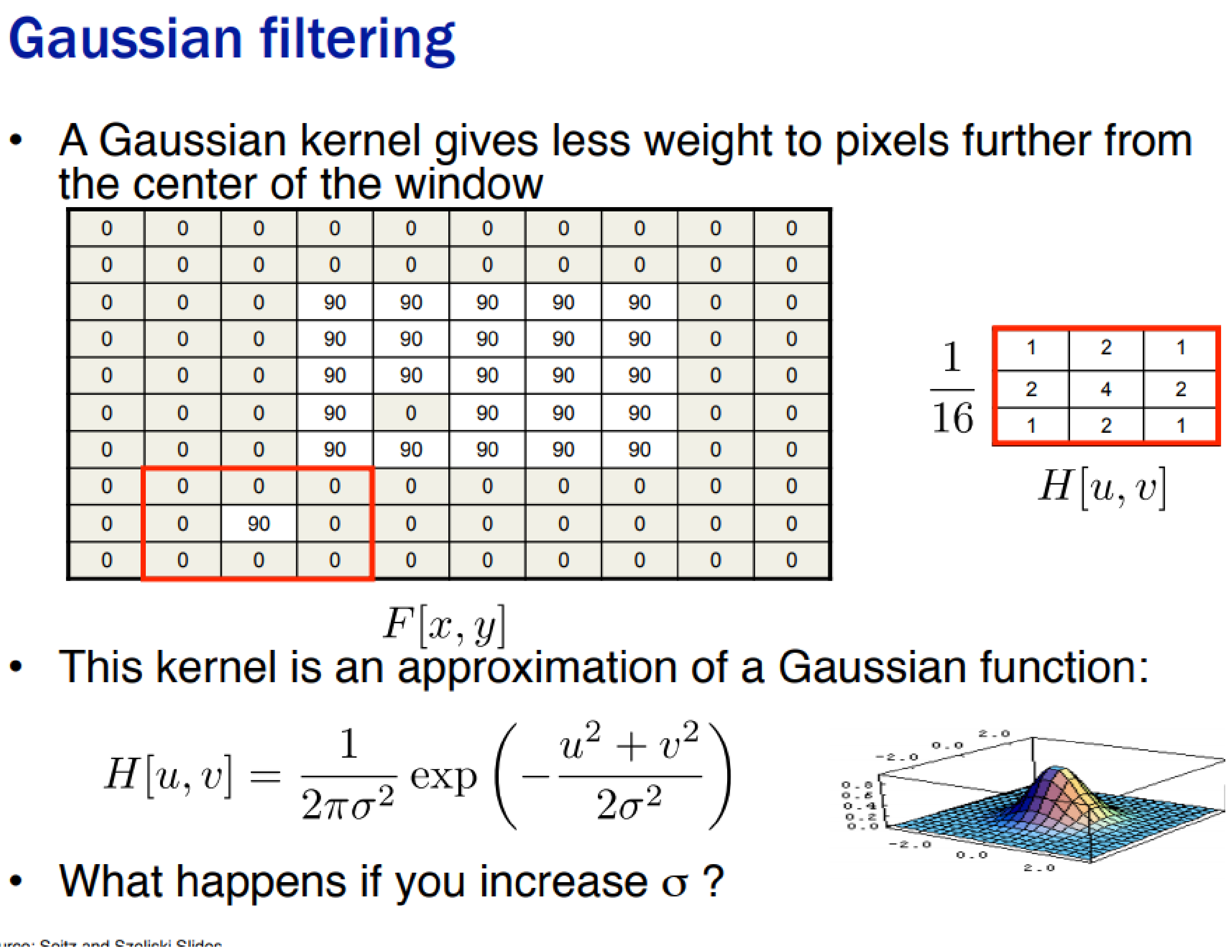
| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 0 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 90 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$F[x,y]$

| | | |
|---|---|---|
| | | |
| | | |
| | | |

$H[u,v]$

- In the case of a **Mean Kernel**, This $H$ will be matrix composed of all $\frac{1}{(2k+1)^2}$.



## Gaussian filtering

- A Gaussian kernel gives less weight to pixels further from the center of the window

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 0 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 90 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$F[x,y]$

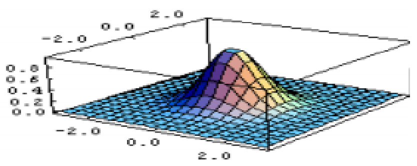
$\frac{1}{16}$

| | | |
|---|---|---|
| 1 | 2 | 1 |
| 2 | 4 | 2 |
| 1 | 2 | 1 |

$H[u,v]$

- This kernel is an approximation of a Gaussian function:

$$H[u,v] = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{u^2+v^2}{2\sigma^2}\right)$$

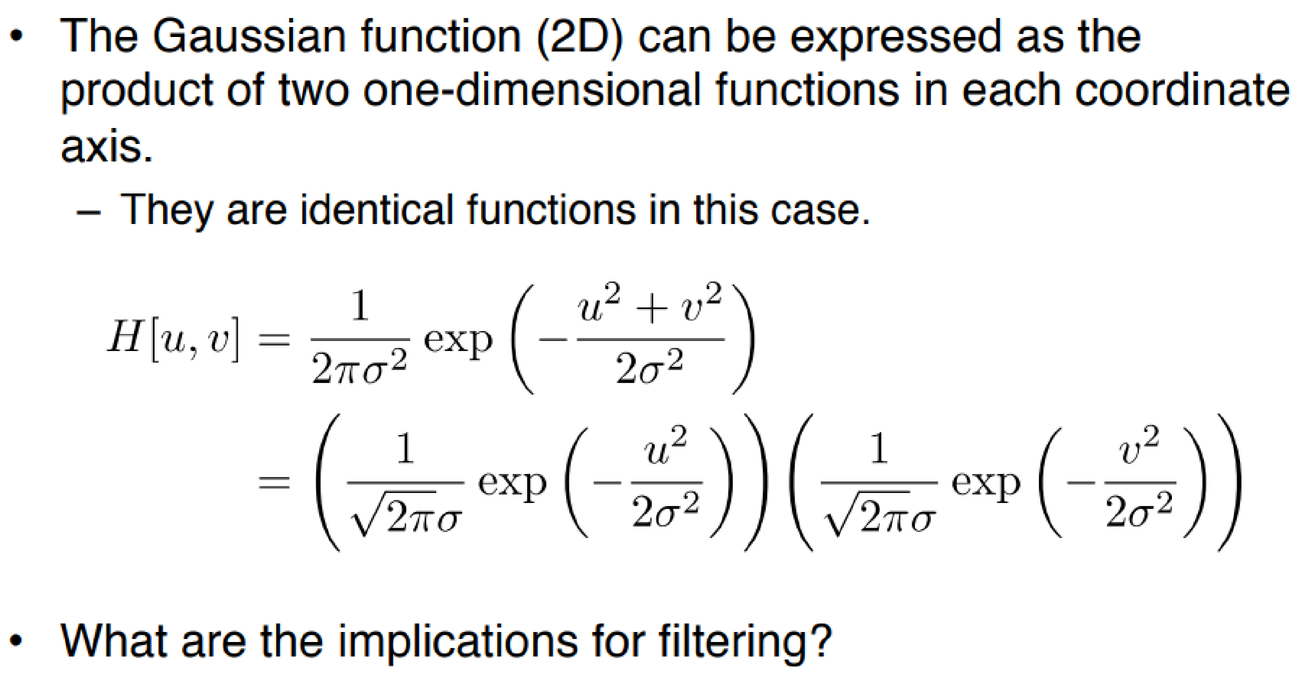


- What happens if you increase σ ?

Source: Seitz and Szeliski Slides

- The more we increase the value of $\sigma$ the less weight will be given to the neighbours.

## Separability of Gaussian Filter



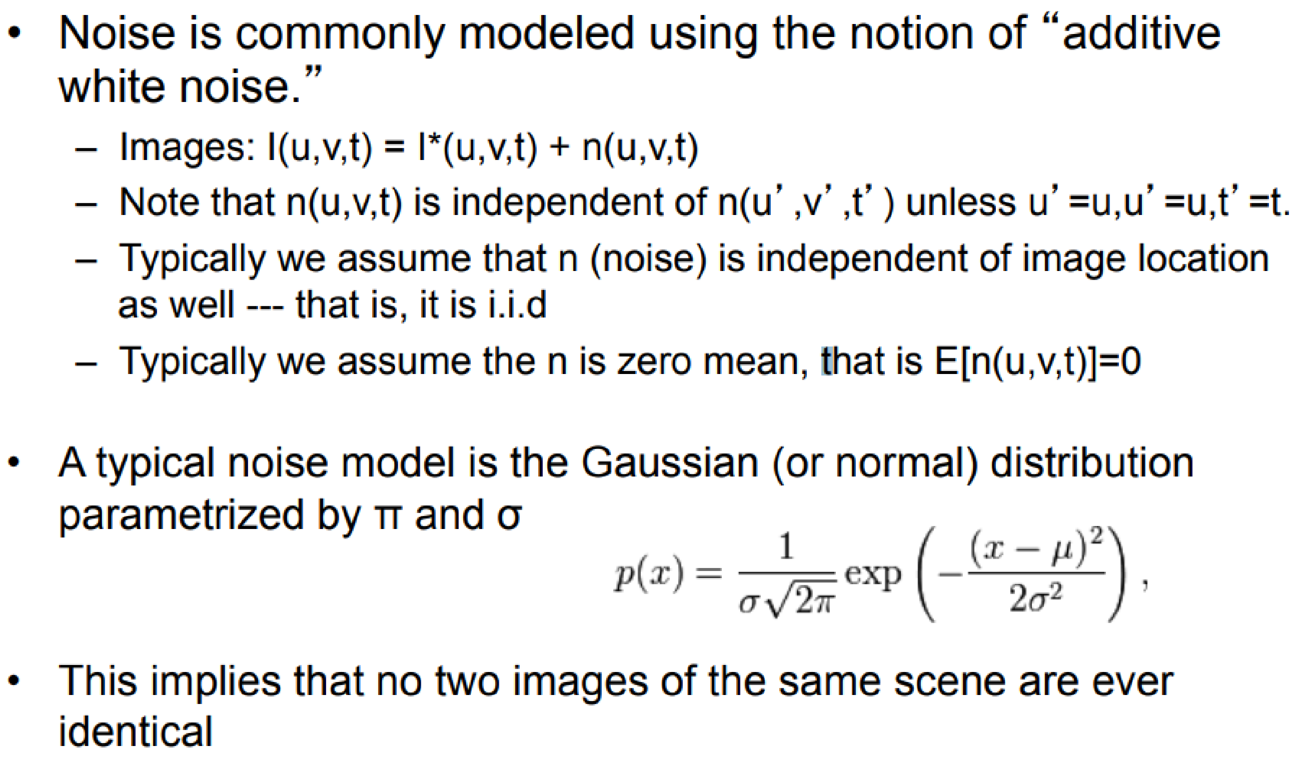
- The Gaussian function (2D) can be expressed as the product of two one-dimensional functions in each coordinate axis.
  - They are identical functions in this case.

$$H[u,v] = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{u^2+v^2}{2\sigma^2}\right)$$
$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{u^2}{2\sigma^2}\right)\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{v^2}{2\sigma^2}\right)\right)$$

- What are the implications for filtering?

-

## Noie Models



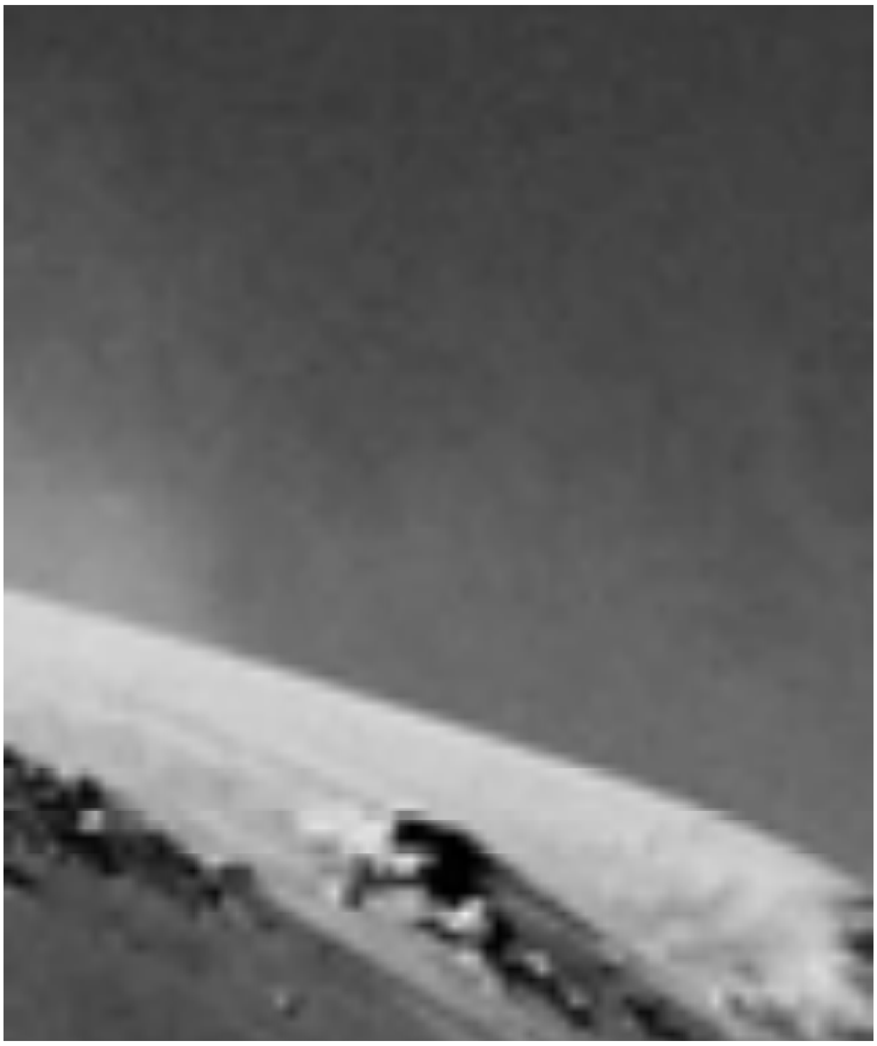
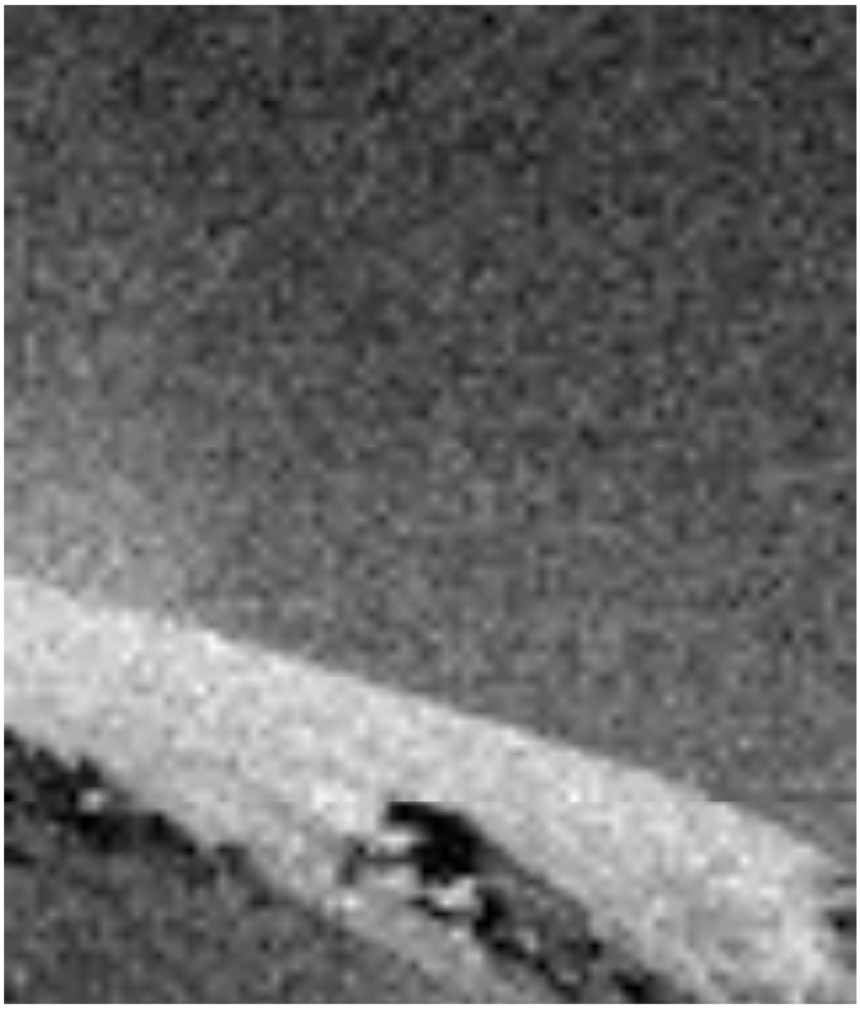
- Noise is commonly modeled using the notion of “additive white noise.”
  - Images: I(u,v,t) = I*(u,v,t) + n(u,v,t)
  - Note that n(u,v,t) is independent of n(u’ ,v’ ,t’ ) unless u’ =u,u’ =u,t’ =t.
  - Typically we assume that n (noise) is independent of image location as well --- that is, it is i.i.d
  - Typically we assume the n is zero mean, that is E[n(u,v,t)]=0
- A typical noise model is the Gaussian (or normal) distribution parametrized by π and σ

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right),$$

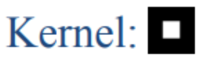
- This implies that no two images of the same scene are ever identical

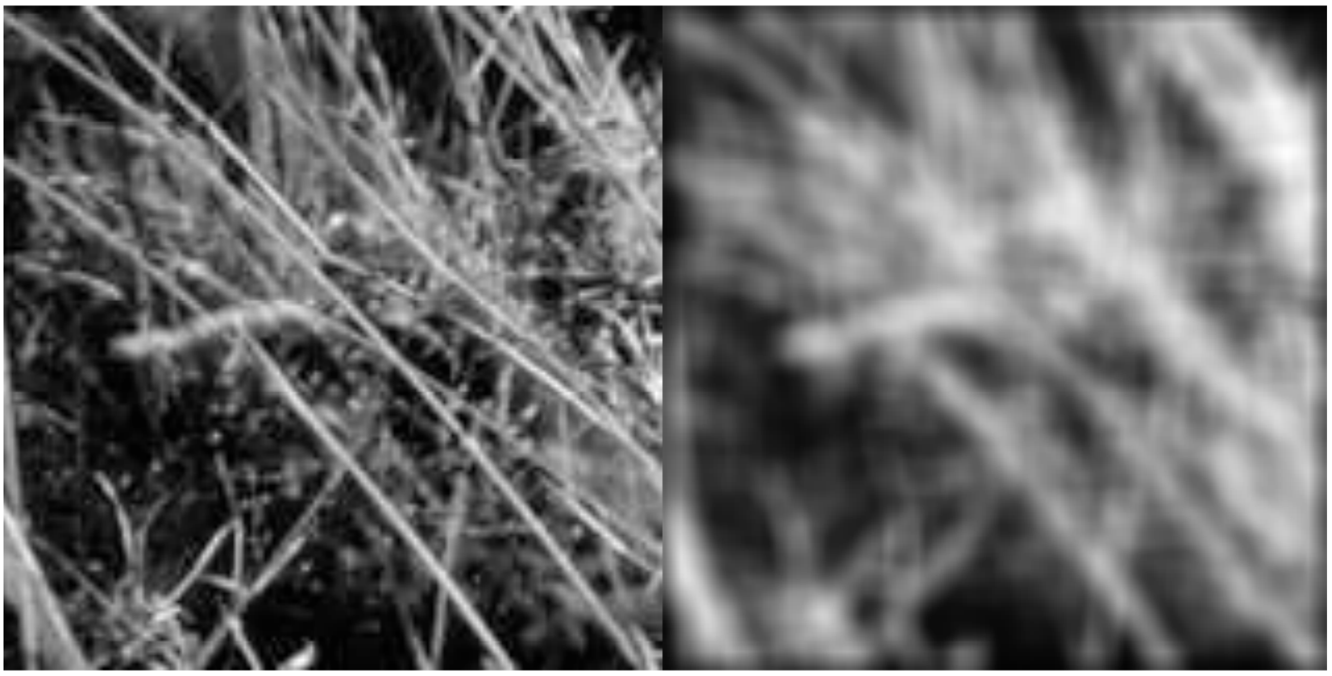
-

- Gaussian Noise is only parametrized by $\pi$ and $\sigma$ as the $\mu = 0$.

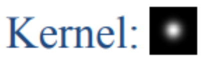
- Gaussian Noise: sigma=1 
- Gaussian Noise: sigma=16 

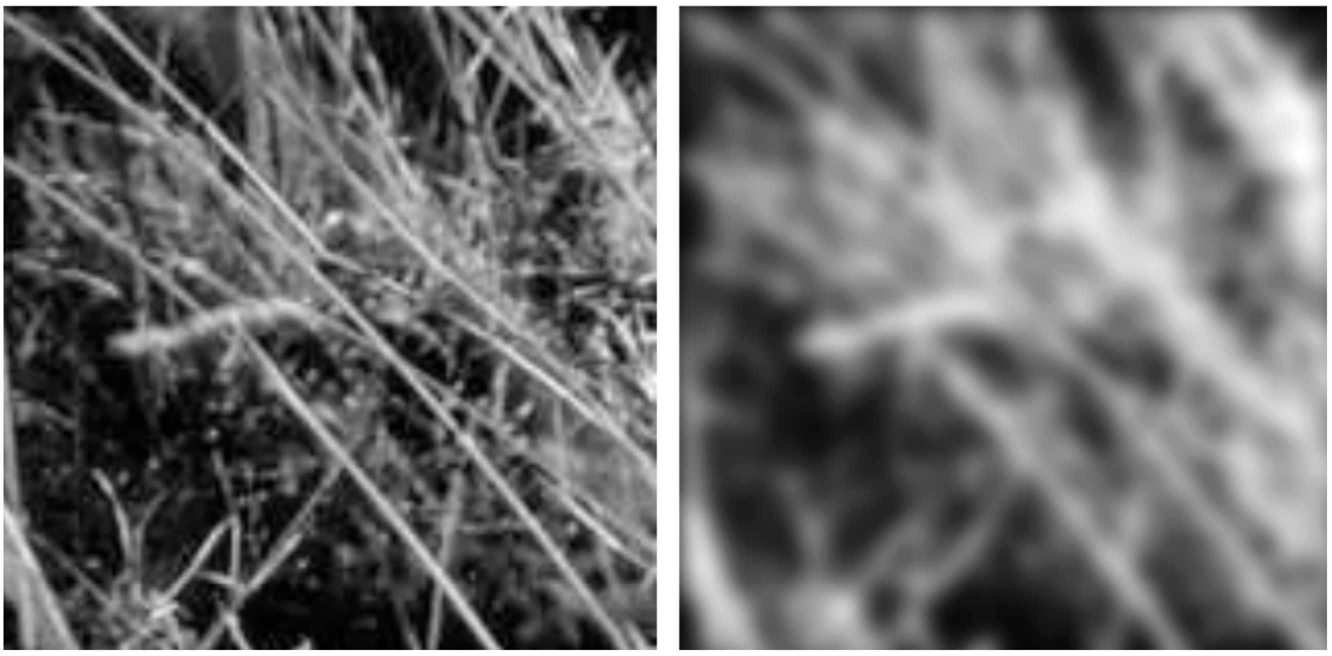
## Smoothing by Averaging

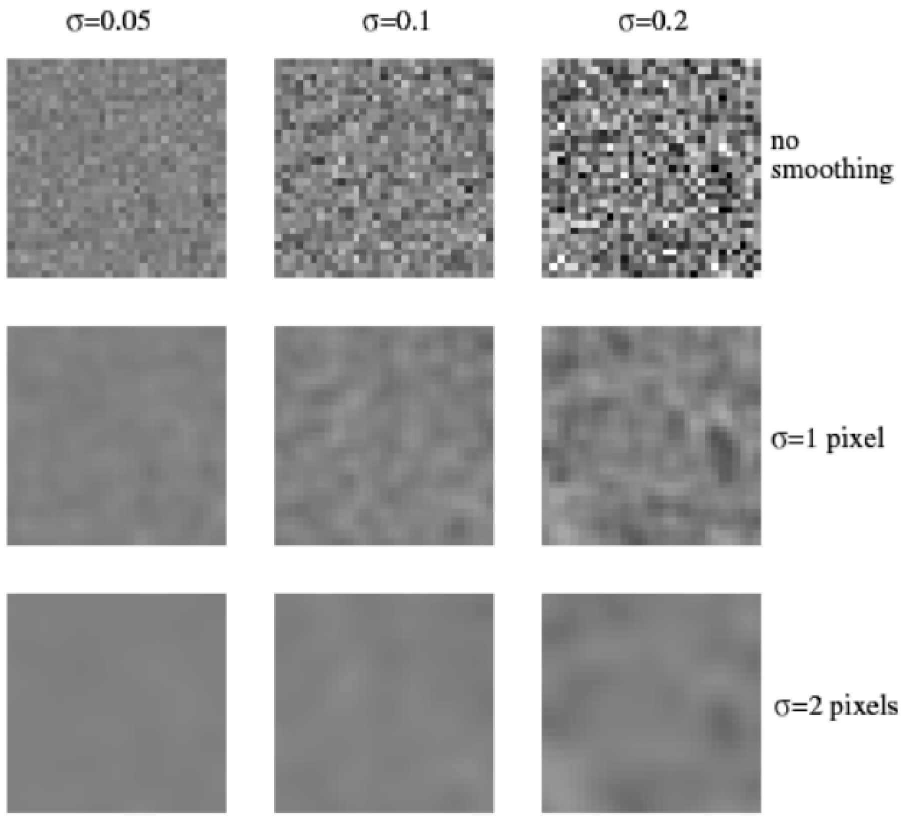


- 

## Smoothing by Gaussian



- 
- look at how the kernal in the case of mean filtering has given equal weights to all the intensity values while in the case of gaussian filter the the weights are assigned as per the gaussian.
-



**The effects of smoothing**
Each row shows smoothing with gaussians of different width; each column shows different realizations of an image of gaussian noise.

## Properties of Noise Processes

- Properties of temporal image noise:

Mean $\mu(i,j) = \Sigma\, I(u,v,t)/n$

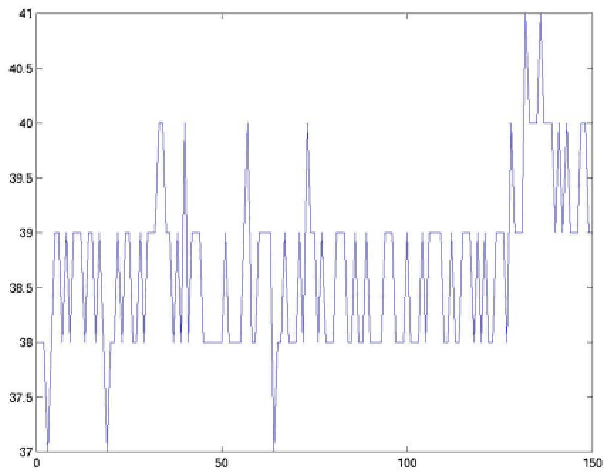
Standard Deviation $\sigma_{i,j} = \mathrm{Sqrt}(\, \Sigma\, (\, \mu(\iota,\varphi) - I(u,v,t)\, )^2/n\ )$

Signal-to-noise Ratio $\dfrac{\mu\,(i,j)}{\sigma_{i,j}}$

•

## Image Noise

- An experiment: take several images of a static scene and look at the pixel values



mean = 38.6
std = 2.99

Snr = 38.6/2.99 ≈ 13
max snr = 255/3 ≈ 85

•

**PROPERTIES OF TEMPORAL IMAGE NOISE**

**(i.e., successive images)**

- If standard deviation of grey values at a pixel is s for a pixel for a single image, then the laws of statistics states that for independent sampling of grey values, for a temporal average of n images, the standard deviation is:

$$\frac{\sigma}{\text{Sqrt}(n)}$$

- For example, if we want to double the signal to noise ratio, we could average 4 images.

•

## Temporal vs. Spatial Noise

- It is common to assume that:
  - spatial noise in an image is consistent with the temporal image noise
  - the spatial noise is independent and identically distributed
- Thus, we can think of a neighborhood of the image itself as approximated by an additive noise process
- Averaging is a common way to reduce noise
  - instead of temporal averaging, how about spatial?
- For example, for a pixel in image I at i,j

$$I'(i,j) = 1/9 \sum_{i'=i-1}^{i+1} \sum_{j'=j-1}^{j+1} I(i',j')$$

•

## Correlation and Convolution

- Correlation: $G = H \otimes F$

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u,j+v]$$

- Convolution: $G = H * F$

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} \underbrace{\boxed{H[u,v]}}_{\text{Impulse Response Function}} F[i-u,j-v]$$

•

## Point Operations

Point Operation is the modification of the pixel value without changing in the size, geometry and local structure of the image. The new pixel value depends only on the previous value. They are mapped by a function f(a)
if the function f() not depend on the coordinate, it is called “global” or “homogeneous” operation. Another one is called “nonhomogeneous” point operation if it depends on the coordinate. Nonhomogeneous point operation is used to compensate for uneven lighting during image acquisition.

The common examples of homogeneous operation include:

- Modifying contrast and brightness
- Limiting the Result by Clamping
- Inverting Image
- Threshold Operation

The implement of the point operation affects on the histogram. Raising the brightness shift the histogram to right and increasing the contrast of the image expand the histogram. These point operations map the intensity by the mapping function contained the constant which is image content such as the highest intensity and the lowest intensity.

## Automatic Contrast Adjustment

Auto-contrast modification is the method to map the lowest intensity and highest intensity which found in the image to the minimum and maximum intensity of the full intensity range respectively (in case of 8 bits gray-scale image, the full range is 0–255). The mapping function of Auto-contrast adjustment is defined as

$$f_{\mathrm{ac}}(a) = a_{\mathrm{min}} + \left(a - a_{\mathrm{low}}\right) \cdot \frac{a_{\mathrm{max}} - a_{\mathrm{min}}}{a_{\mathrm{high}} - a_{\mathrm{low}}},$$

From the equation, a min, a max, a low and a high are minimum value intensity, maximum value intensity in the range, lowest intensity and highest intensity respectively. In case of 8-bit image, a min=0 and a max=255 so the mapping function is defined as

$$f_{\mathrm{ac}}(a) = \left(a - a_{\mathrm{low}}\right) \cdot \frac{255}{a_{\mathrm{high}} - a_{\mathrm{low}}}.$$

-

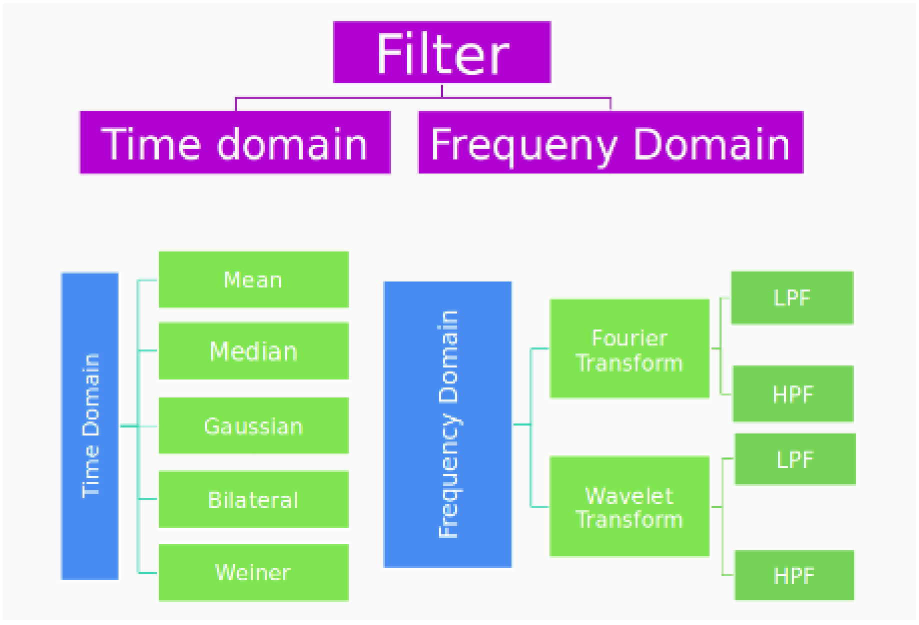
## Modify Auto-Contrast

The highest intensity and the lowest intensity may be the noise of the image, We exclude these noise by saturating intensity of the image using quantile. And it can be calculated from the equation below.

$$a'_{\text{low}} = \min\{\, i \mid \mathsf{H}(i) \geq M \cdot N \cdot q_{\text{low}} \},$$
$$a'_{\text{high}} = \max\{\, i \mid \mathsf{H}(i) \leq M \cdot N \cdot (1 - q_{\text{high}}) \},$$

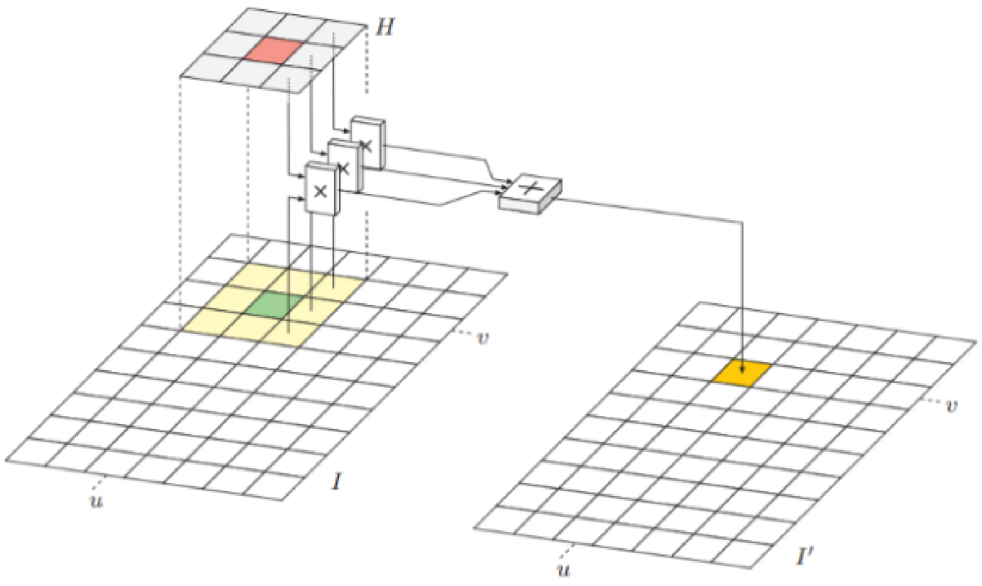
And the mapping equation is defined as

$$f_{\text{mac}}(a) = \begin{cases} a_{\min} & \text{for } \; a \leq a'_{\text{low}} \\ a_{\min} + (a - a'_{\text{low}}) \cdot \dfrac{a_{\max} - a_{\min}}{a'_{\text{high}} - a'_{\text{low}}} & \text{for } \; a'_{\text{low}} < a < a'_{\text{high}} \\ a_{\max} & \text{for } \; a \geq a'_{\text{high}}. \end{cases}$$

- 
- Further Reading:
- https://towardsdatascience.com/image-processing-class-egbe443-3-point-operation-477ad38334f5
- 

## Linear filter

Linear filter is a filter which operate the pixel value in the support region in linear manner (i.e.,as weighted summation). The support region is specified by the ***'filter matrix'*** and be represent as ***H(i,j)***. The size of H is call 'filter region' and filter matrix has its own coordinate system, i is column index and j is row index. The center of it is the origin location and it is called the 'hot spot'.



Applying weight median filter to the image I, a hotspot location is at the orange shade (center of the filter matrix H)

-

**Applying the filter**

To apply the filter to the image, please follow these step.

- Move the filter matrix over the image I and H(0,0) must go along with the current image position (u,v)
- Multiply each filter coefficient H(i,j) with the corresponding image element I(u+i,v+j)
- Average all result from the previous step and it is the result for the current location I(u,v)

All steps can be described as equation below

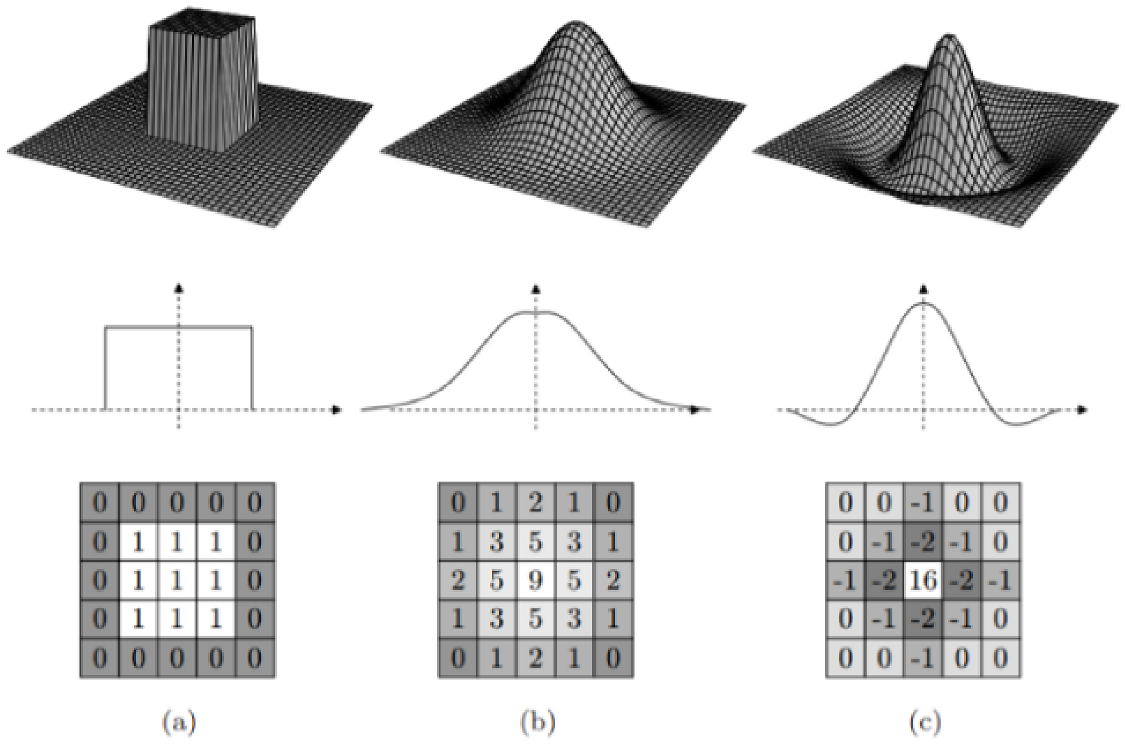
$$I'(u,v) \leftarrow \sum_{i=-1}^{i=1} \sum_{j=-1}^{j=1} I(u+i, v+j) \cdot H(i,j),$$

**Type of linear filter**

1. *Smoothing Filter* (This filter has only positive integer.)

- ***Box filter.*** All members of this filter are the same.
- ***Gaussian filter.*** The weight of filter member depend on the location of the member. The center of the filter receive the maximum weigh and it decreases with distance from the center.
-

- ***Laplace or Mexican hat filter.*** Some members of this filter are negative filter and it can calculate by summation of positive member and negative member.



|  |  |  |  |  |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 |

|  |  |  |  |  |
|---|---|---|---|---|
| 0 | 1 | 2 | 1 | 0 |
| 1 | 3 | 5 | 3 | 1 |
| 2 | 5 | 9 | 5 | 2 |
| 1 | 3 | 5 | 3 | 1 |
| 0 | 1 | 2 | 1 | 0 |

|  |  |  |  |  |
|---|---|---|---|---|
| 0 | 0 | -1 | 0 | 0 |
| 0 | -1 | -2 | -1 | 0 |
| -1 | -2 | 16 | -2 | -1 |
| 0 | -1 | -2 | -1 | 0 |
| 0 | 0 | -1 | 0 | 0 |

3D structure, 2D structure and example of filter (a) Box filter (b) Gaussian filter and (c) Laplace filter

-

Monsieur Laplace came up with this equation. This is simply the definition of the Laplace operator: the sum of second order derivatives (you can also see it as the trace of the Hessian matrix).

The second equation you show is the finite difference approximation to a second derivative. It is the simplest approximation you can make for discrete (sampled) data. The derivative is defined as the slope (equation from Wikipedia):

$$f'(a) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h}.$$

In a discrete grid, the smallest `h` is 1. Thus the derivative is `f(x+1)-f(x)`. This derivative, because it uses the pixel at `x` and the one to the right, introduces a half-pixel shift (i.e. you compute the slope in between these two pixels). To get to the 2nd order derivative, simply compute the derivative on the result of the derivative:

```
f'(x) = f(x+1) - f(x)
f'(x+1) = f(x+2) - f(x+1)

f"(x) = f'(x+1) - f'(x)
      = f(x+2) - f(x+1) - f(x+1) + f(x)
      = f(x+2) - 2*f(x+1) + f(x)
```

Because each derivative introduces a half-pixel shift, the 2nd order derivative ends up with a 1-pixel shift. So we can shift the output left by one pixel, leading to no bias. This leads to the sequence `f(x+1)-2*f(x)+f(x-1)`.

Computing this 2nd order derivative is the same as convolving with a filter `[1,-2,1]`.

Applying this filter, and also its transposed, and adding the results, is equivalent to convolving with the kernel

```
[ 0, 1, 0        [ 0, 0, 0        [ 0, 1, 0
  1,-4, 1    =     1,-2, 1    +     0,-2, 0
  0, 1, 0 ]        0, 0, 0 ]        0, 1, 0 ]
```

•

## Properties of Linear Filter

First, i will introduce an operation which associate with linear filter. This operation is call ***"Linear Convolution".*** For two-dimensional function I and H, the convolution operation is defined as the equation

$$I'(u,v) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} I(u-i, v-j) \cdot H(i,j),$$

$$I' = I * H$$

where * is the convolution operation. Look at the equation you will see that this operation provide the similar result with the linear filter with the filter function which reflect in both horizontal and vertical axis. The convolution matrix H can be called ***kernel***.

**Properties of Linear Convolution**

- Commutativity
- Linearity
- Associativity
- Seperability: the kernel H can be represented as the convolution of multiple kernels and can separated in a pair dimensional kernel x and y.

***Note that****: In the linearity properties, adding scalar value b to the image I before perform convolution with the kernel dose not equal to adding scalar value b to convolution result between the image and the kernel.*

-

## Non-Linear Filters

Noise removing with smoothing filter (a linear filter) provide the result in burred of the image structure, line and edge. Non-Linear Filters were used to solve this problem and it works in non-linear manner.

### Type of non-linear filters

- ***Minimum and Maximum Filters:*** The minimum and maximum value in the moving region R of the original image is the result of the minimum and maximum filter respectively. These filter were defined as

$$I'(u,v) \leftarrow \min\{I(u+i, v+j) \mid (i,j) \in R\},$$
$$I'(u,v) \leftarrow \max\{I(u+i, v+j) \mid (i,j) \in R\},$$

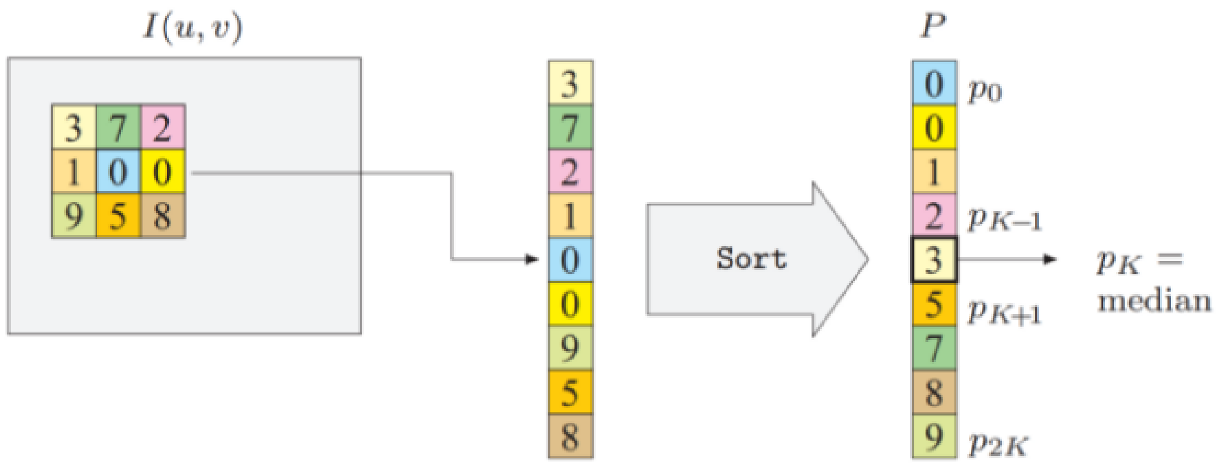
The equation of minimum and maximum filter

-

- ***Median Filter:*** The result was calculated in the same way as the minimum and maximum filter. The median of all value in moving region R is the result of the median filter. And this filter typically use for remove salt and pepper noise in the image. This filter was defined as

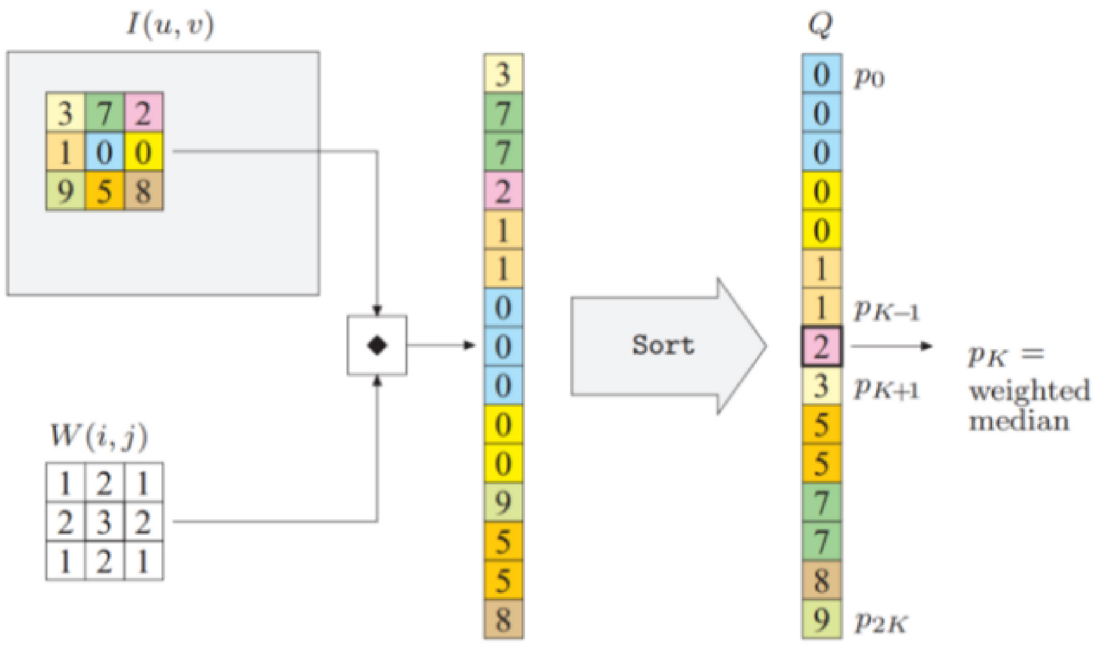
$$I'(u,v) \leftarrow \text{median}\,\{I(u+i, v+j) \mid (i,j) \in R\}\,.$$

and the figure below show how the median filter work



-

- ***Weight Median Filter:*** This similar to the median filter, The word "Weight" means extension of each member in the kernel according to weight matrix W. The position of the filter member near hot spot will obtain the higher weight than others.



-
-