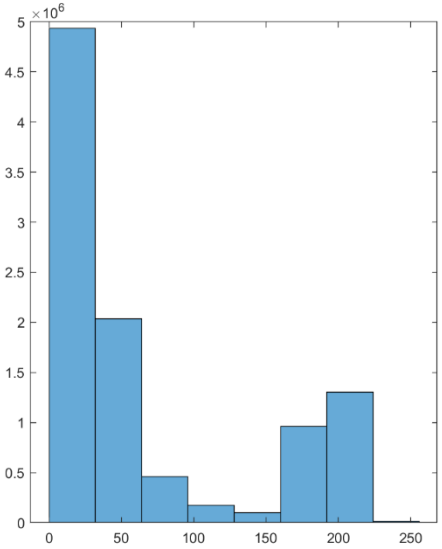
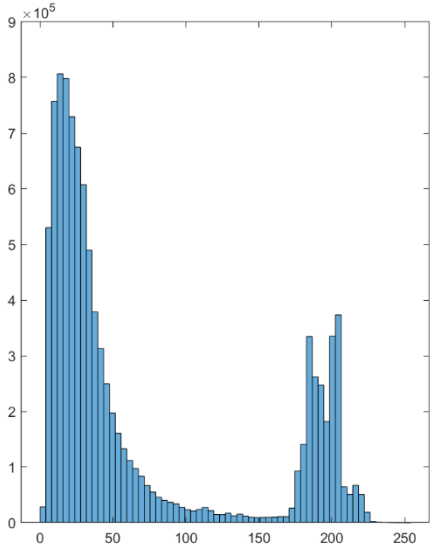
Histogram

Bin	Range	No. of samples within the range	Probability
0	0-31	4932982	49.4%
1	32-63	2033968	20.3%
2	64-95	459607	4.6%
3	96-127	175259	1.8%
4	128-159	98157	1.0%
5	160-191	960309	9.6%
6	192-223	1304856	13.1%
7	224-255	15790	0.2%
Total		9980928	100%



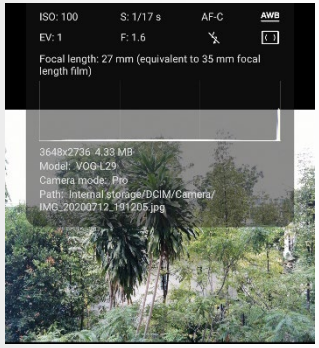


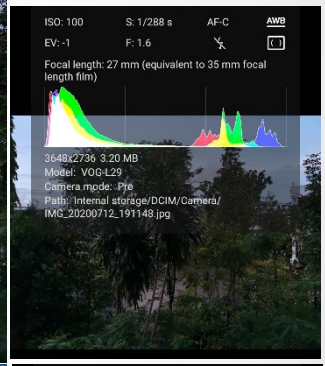
Data: [2736×3648 uint8]
Values: [4932982 2033968 459607 175259 98157 960309 1304856 15790]
NumBins: 8
BinEdges: [0 32 64 96 128 160 192 224 256]
BinWidth: 32

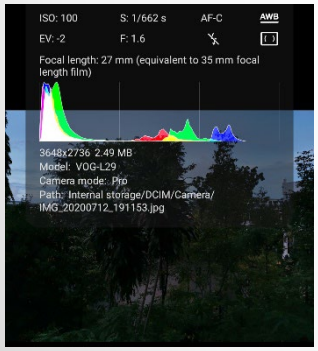
NumBins: 64

Histogram

- Cameras can show real-time histogram of photo.
- It helps avoid taking over-exposed photos.

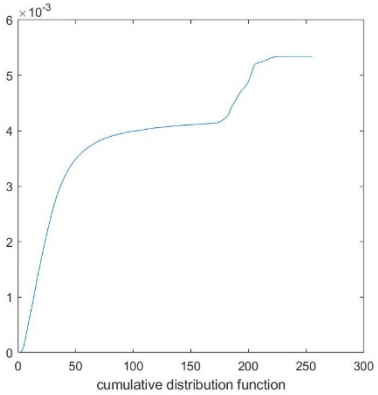


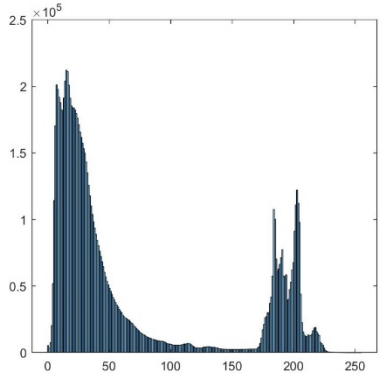


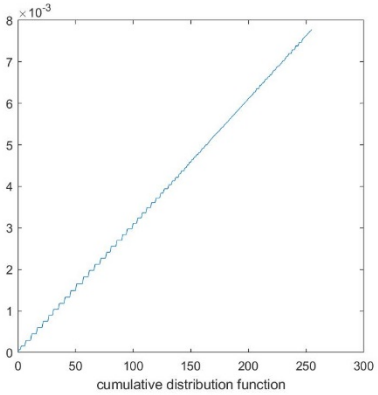


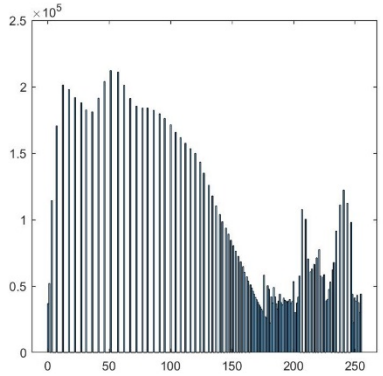
Histogram equalization

- Histogram equalization is a technique by which the distribution of an image is changed to obtain a uniform resulting histogram.







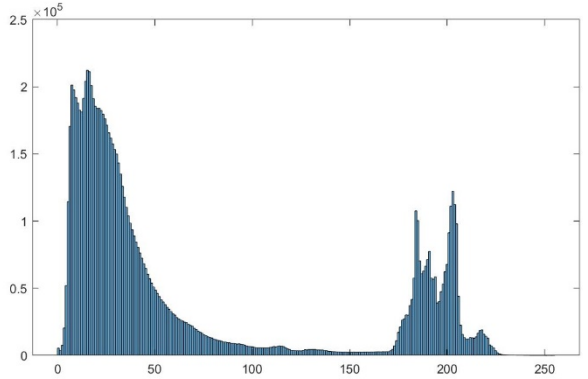


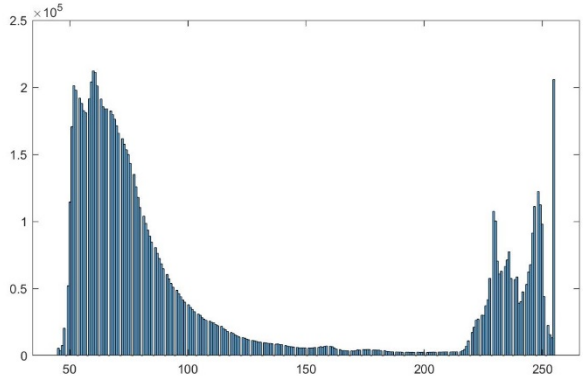
Histogram equalization

Color version of previous histogram equalization

Histogram sliding

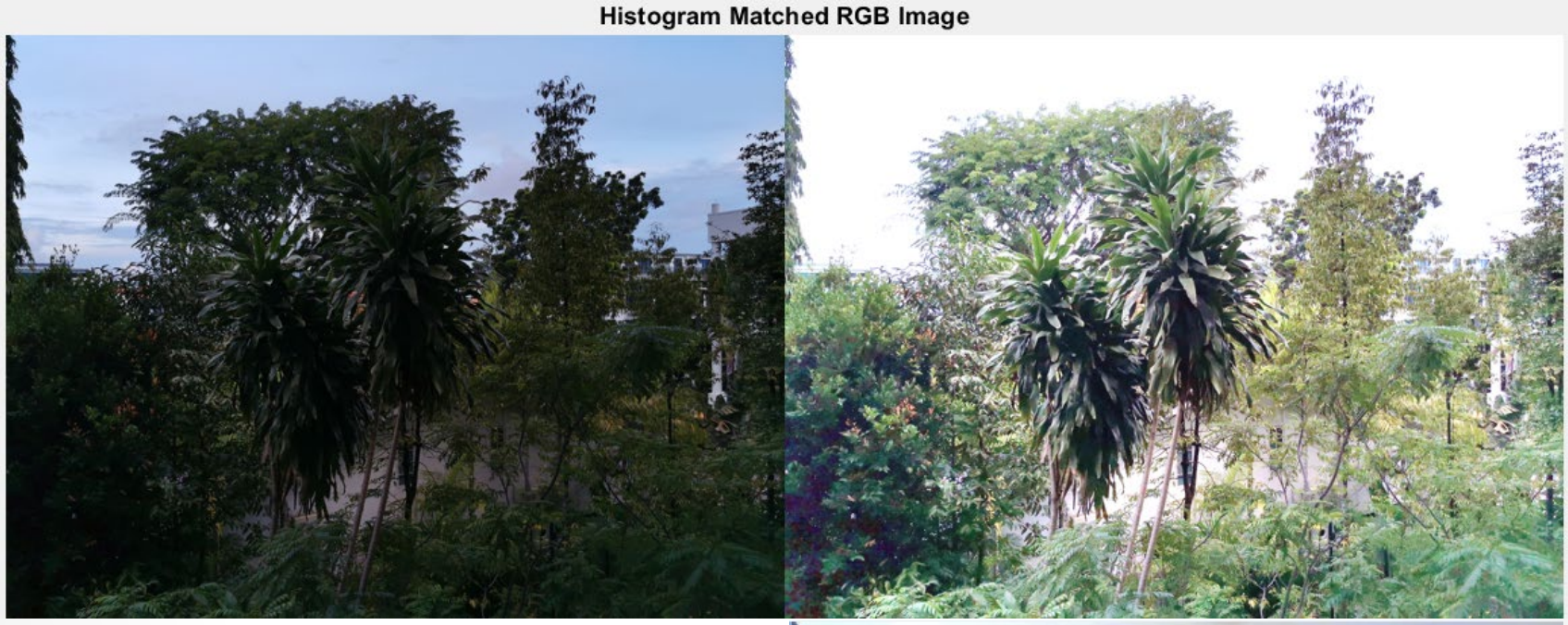
- By adding or subtracting a constant to all pixels in the image, The effect will be increasing or decreasing brightness.

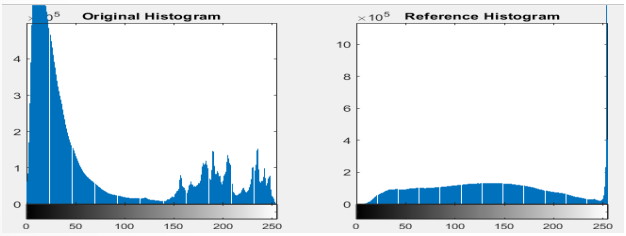


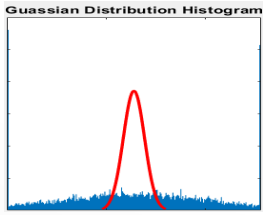


Slide the histogram to the right side of 45, and clip at 255.

Histogram filter



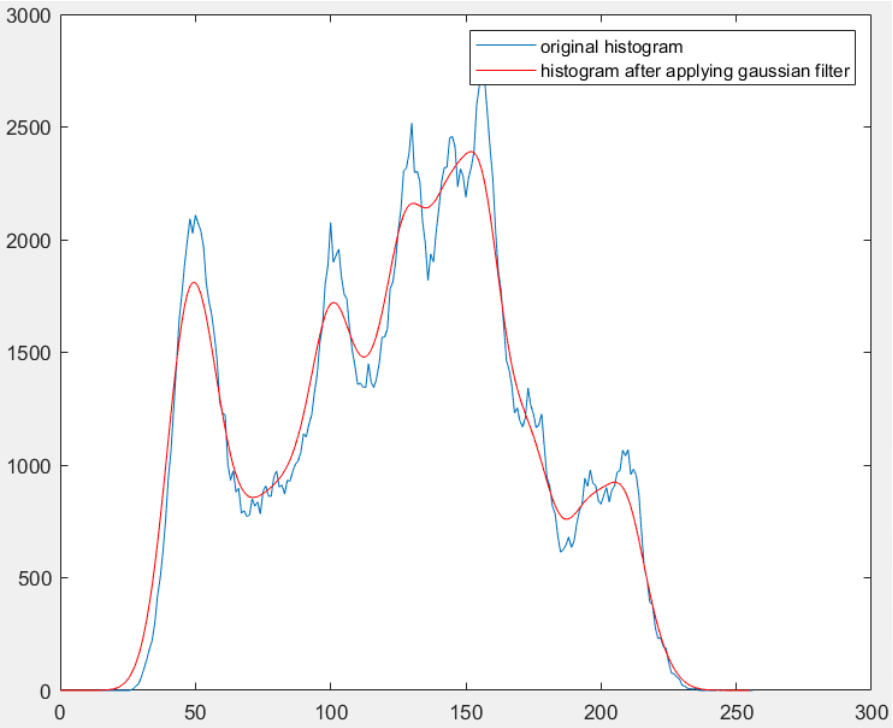




Applying Gaussian distribution to histogram signal

Histogram filtering with Gaussian filter

- Preserving the mean brightness of the original image in the output image.
- Pixel values distribution more even.



Histogram filtering

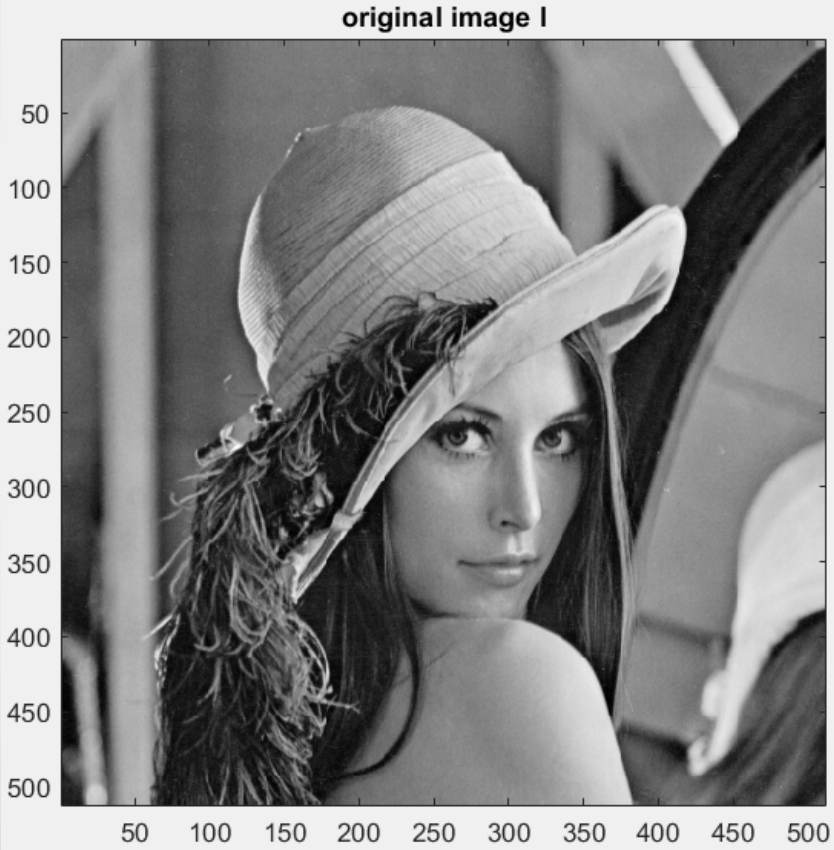


Image filter

Operations:

1. Define a reference pixel in the input image, I(x,y).
2. Perform a filtering that involves only pixels around the reference pixel in image I.
3. Apply the filtering result to the pixel of coordinate (x,y) in the output image, J(x,y).
4. Repeat the process for every pixel in the image I.

Two categories of image filter:

1. Linear filters: the value of an output pixel is a linear combination of the values of the pixels in the input pixel's neighborhood. Example: mean filter.
2. Nonlinear Filters: the value of an output pixel is selected from an ordered sequence of the values of the pixels in the input pixel's neighborhood. Example: median filter

Image Convolution

- Convolution is a widely used mathematical operator provides a way to multiply two arrays of numbers. It can be different size, but of the same dimensionality.
- Image convolution processes an image by computing a weighted sum of the values of each pixel and its neighbors.
- Mathematical definition
  - 1D: $A * B = \sum_{j=-\infty}^{\infty} A(j) \cdot B(x-j)$ for pixel (x)
  - 2D: $A * B = \sum_{k=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} A(j,k) \cdot B(x-j, y-k)$ for pixel (x,y)

1D Convolution operation

Signal:

A = {2,0,2,3,1}

Kernal:

B = {1,2,−1}

Flip array B and slide along the axis

Result:

3 values, {0, 7, 5}

A			2	0	2	3	1		
B	-1	2	1						
A*B		2							
A			2	0	2	3	1		
B		-1	2	1					
A*B		2	4						
A			2	0	2	3	1		
B			-1	2	1				
A*B		2	4	0					
A			2	0	2	3	1		
B				-1	2	1			
A*B		2	4	0	7				
A			2	0	2	3	1		
B					-1	2	1		
A*B		2	4	0	7	5			
A			2	0	2	3	1		
B						-1	2	1	
A*B		2	4	0	7	5	-1		
A			2	0	2	3	1		
B							-1	2	1
A*B		2	4	0	7	5	-1	-1	

convolution

Convolution or Correlation?

- **Convolution** is measurement of effect of one signal on the other signal. Convolution in time domain equals multiplication in frequency domain.

$$A * B = \sum_{j=-\infty}^{\infty} A(j) \cdot B(x - j)$$

- **Correlation** is measurement of the similarity between two signals.

$$A * B = \sum_{j=-\infty}^{\infty} A(j) \cdot B(x + j)$$

Image filter

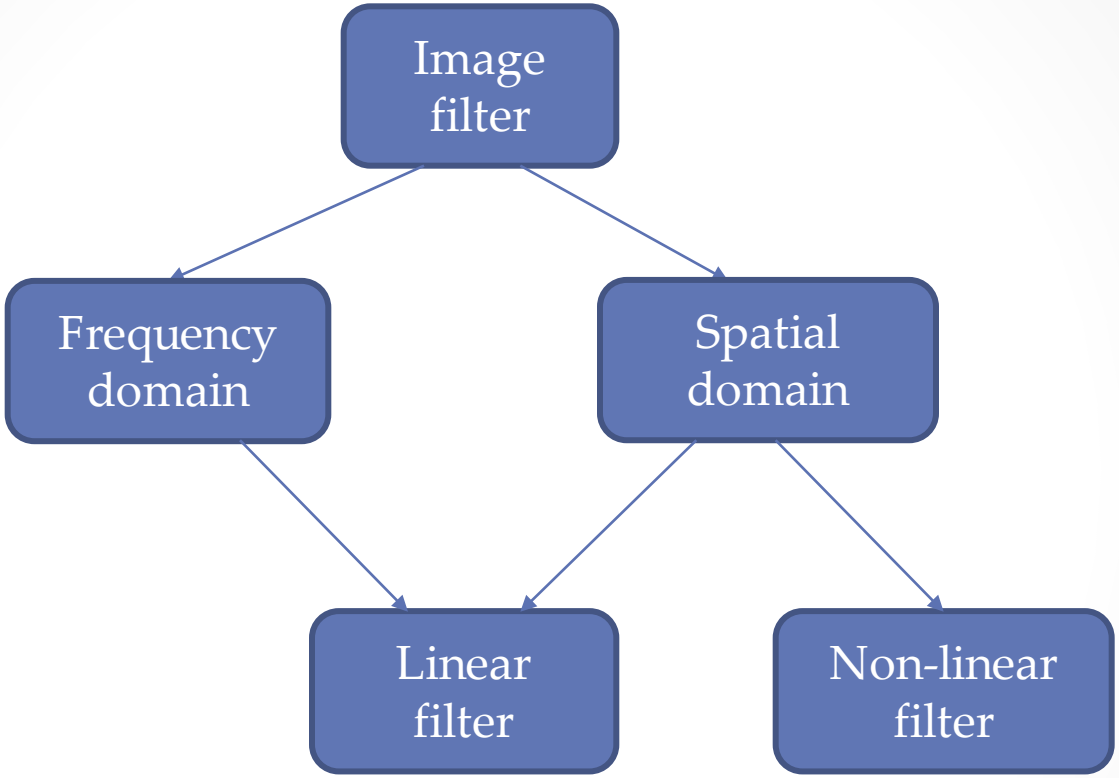
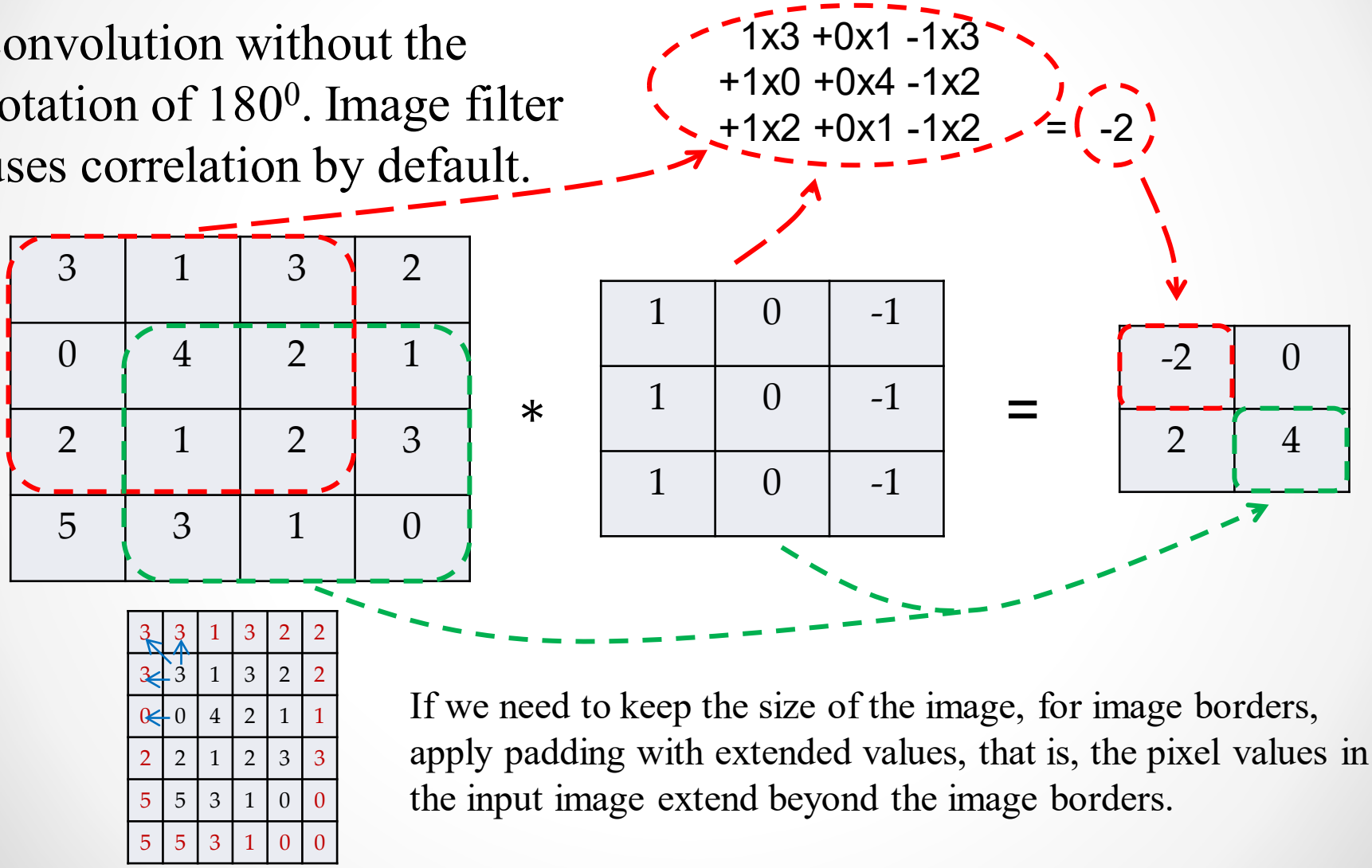


Image (Spatial) Filtering

Correlation is the same as convolution without the rotation of 180^0. Image filter uses correlation by default.



If we need to keep the size of the image, for image borders, apply padding with extended values, that is, the pixel values in the input image extend beyond the image borders.

Spatial domain linear filter

- Use 3x3 kernel (mask) as example

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Mean (low pass) filter

$$\begin{bmatrix}0 & -1 & 0\\ -1 & 4 & -1\\ 0 & -1 & 0\end{bmatrix}$$

High pass filter 1

$$\begin{bmatrix}1 & 0 & -1\\ 1 & 0 & -1\\ 1 & 0 & -1\end{bmatrix}$$

Vertical edge detector

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Approximated Gaussian kernels 3x3 (low pass) filter

$$\begin{bmatrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{bmatrix}$$

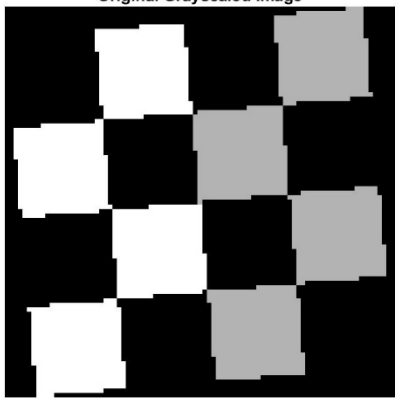
High pass filter 2

Image filter

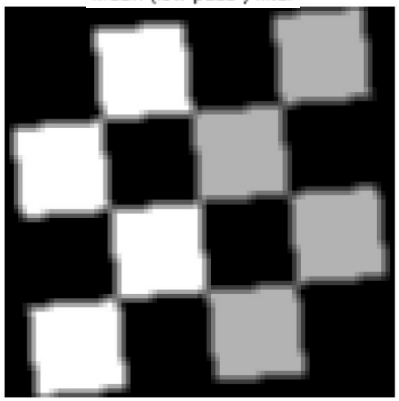
$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

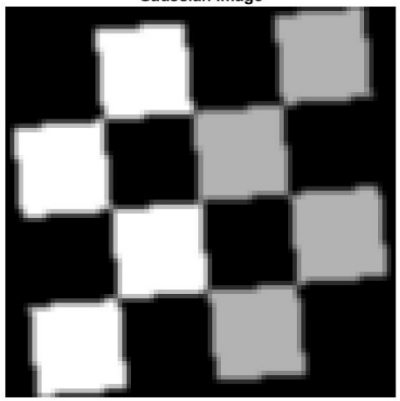
Original Grayscaled Image



Mean (low pass) filter

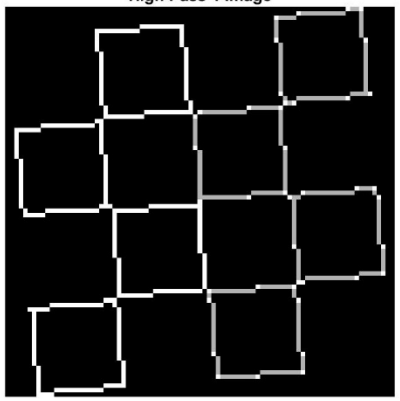


Gaussian Image



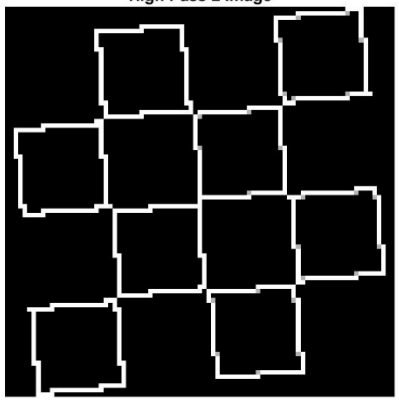
$$\begin{bmatrix}0 & -1 & 0\\ -1 & 4 & -1\\ 0 & -1 & 0\end{bmatrix}$$

High Pass 1 Image



$$\begin{bmatrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{bmatrix}$$

High Pass 2 Image



$$\begin{bmatrix}1 & 0 & -1\\ 1 & 0 & -1\\ 1 & 0 & -1\end{bmatrix}$$

Vertical Edge Image

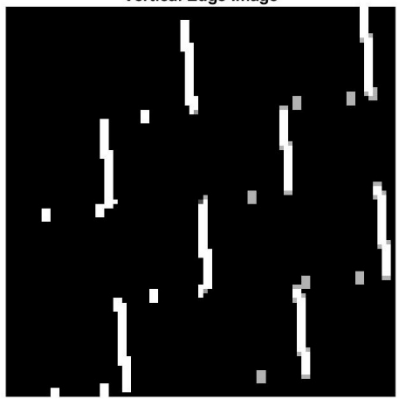


Image filter

Original Grayscaled Image

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

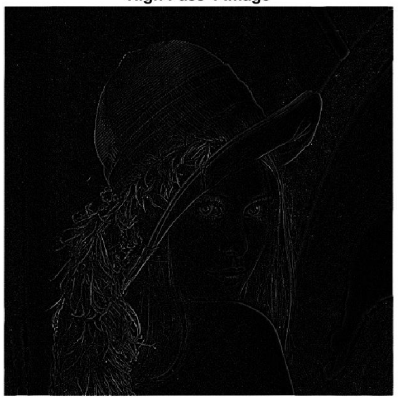
Low Pass Image

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Gaussian Image

$$\begin{bmatrix}0 & -1 & 0\\ -1 & 4 & -1\\ 0 & -1 & 0\end{bmatrix}$$

High Pass 1 Image



$$\begin{bmatrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{bmatrix}$$

High Pass 2 Image

$$\begin{bmatrix}1 & 0 & -1\\ 1 & 0 & -1\\ 1 & 0 & -1\end{bmatrix}$$

Vertical Edge Image

Low pass (Mean) filter with different kernel size

Low Pass Image 3x3

Low Pass Image 7x7

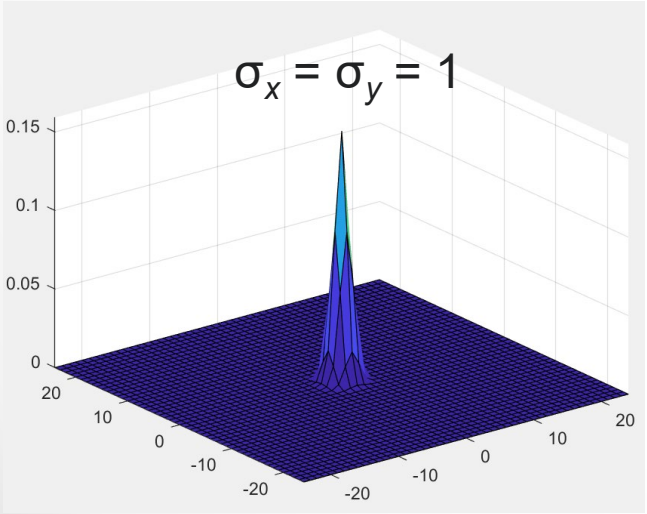
Gaussian filter

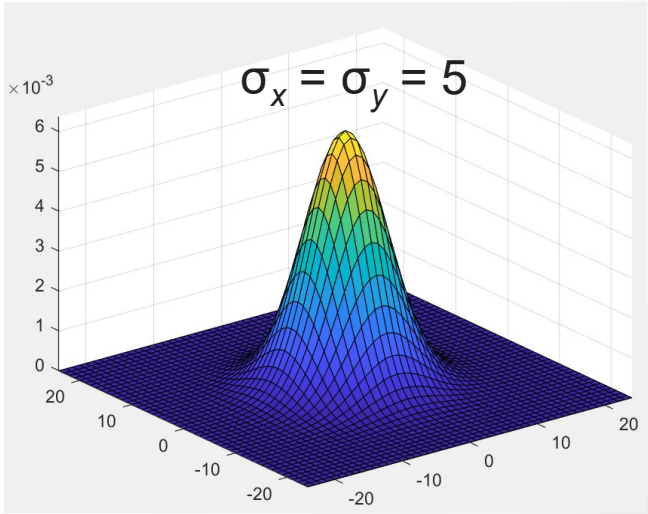
An example of a 2D Gaussian function is in the following.

$$g(x,y) = Ae^{-\left(\frac{(x-x_0)^2}{2\sigma_x^2}+\frac{(y-y_0)^2}{2\sigma_y^2}\right)}$$

where A is amplitude, x_o , y_o is the centre, $\sigma_x = \sigma_y$ is variance.

A = 1, x_o = 0, y_o = 0.





Gaussian filter

Median filter

The median filter works well in removing salt and pepper noise from image. Salt => bright, pepper => dark.

e.g. input {1,10, 1}

mean filter (1+10+1)/3=4, median filter {1,1,10}=1

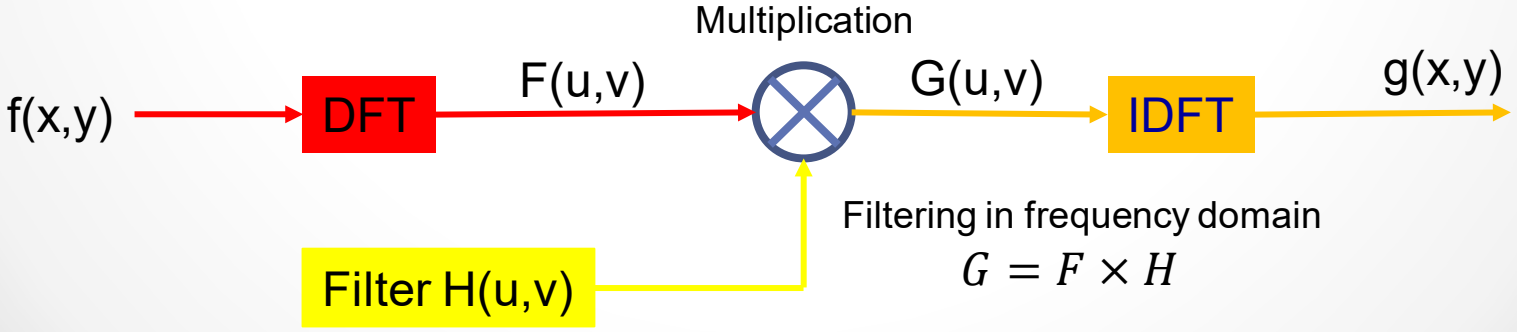
Original | Add salt and pepper noise | Apply 3x3 median filter

Frequency domain filter of spatial filter

- 2-D DFT and IDFT for NxN image pixels is given as

$$F(u,v) = \frac{1}{N*N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \mathrm{e}^{-j2\pi\left(\frac{ux+vy}{N}\right)} \qquad \text{for u,v = 0,1,…,N-1}$$

$$g(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} G(u,v) \mathrm{e}^{j2\pi\left(\frac{ux+vy}{N}\right)} \qquad \text{for x,y = 0,1,…,N-1}$$



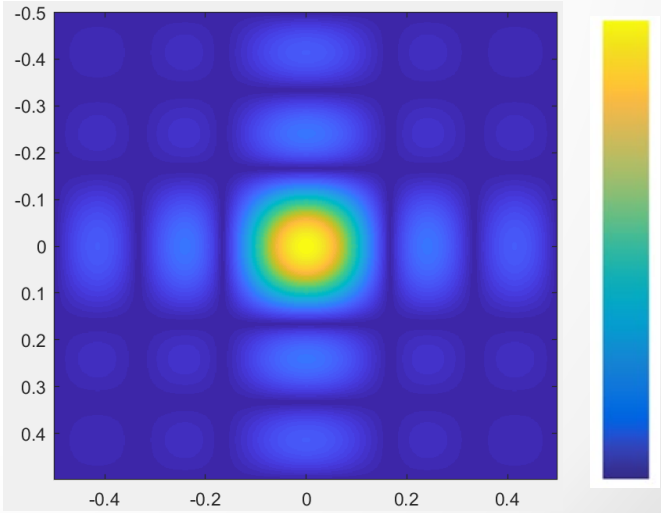
Frequency domain filter of spatial filter

$$\frac{1}{36}\times\begin{pmatrix}1&1&1&1&1&1\\1&1&1&1&1&1\\1&1&1&1&1&1\\1&1&1&1&1&1\\1&1&1&1&1&1\\1&1&1&1&1&1\end{pmatrix}$$

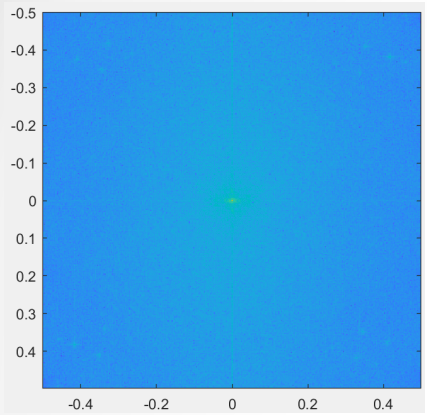
Zero padding to fit the size of 512x512

$$\frac{1}{36}\begin{pmatrix}1&1&1&1&1&1&&\\1&1&1&1&1&1&&\\1&1&1&1&1&1&\cdots&0\\1&1&1&1&1&1&&\\1&1&1&1&1&1&&\\1&1&1&1&1&1&&\\&&\vdots&&&&\ddots&\vdots\\&&0&&&&\cdots&0\end{pmatrix}$$

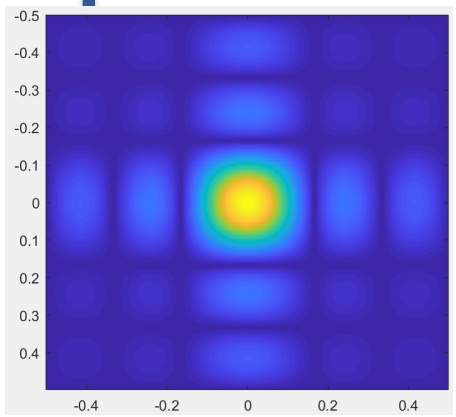
Apply 2-D DFT



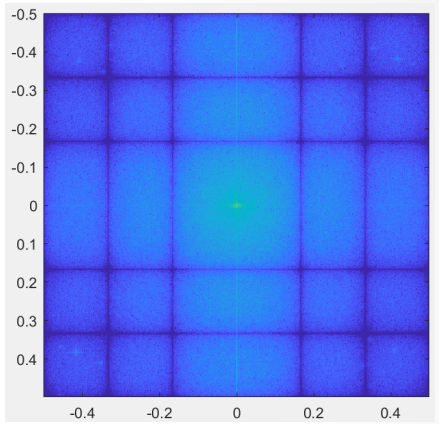
Frequency domain filter of spatial filter



X



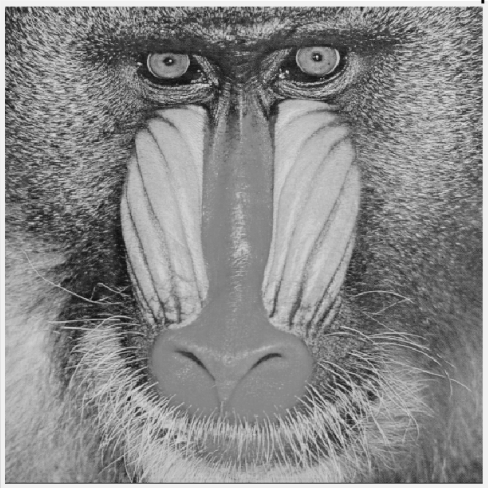
=

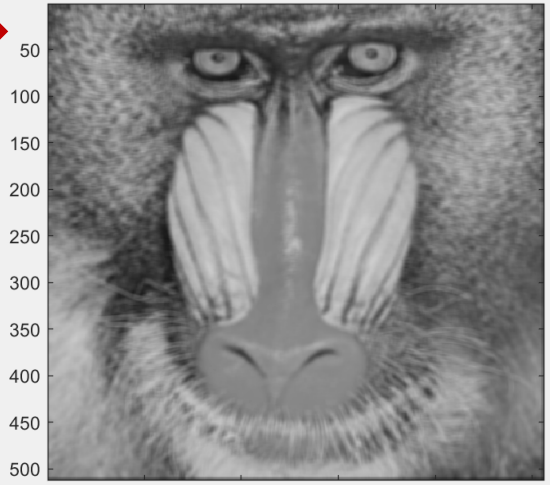


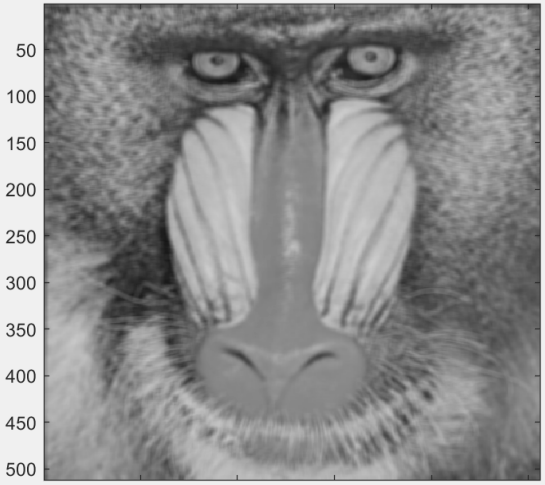
Apply 2-D DFT

Apply 6x6 spatial filter in spatial domain

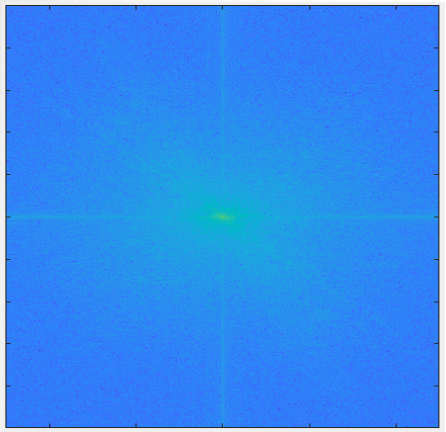
Apply 2-D IDFT







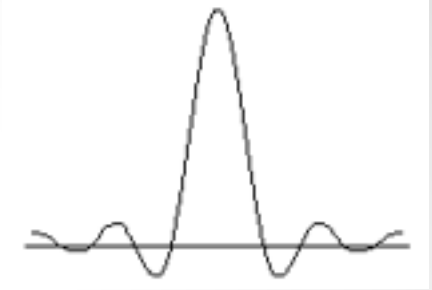
Frequency domain filter of spatial filter



Same method to apply for Lena

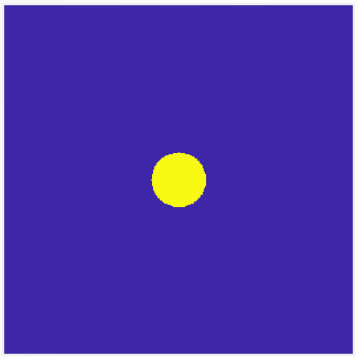
Ideal low pass filter

- The transfer function of the ideal lowpass filter has ripple shape in spatial domain.
- When apply convolution in spatial domain, this ripple will carry a few pixels away pixel intensity to the center pixel. If the intensity difference is large (a sharp transition or an edge), we will see ringing effect.
- Ringing effect appeared as rippling artifact near sharp edges. This effect is caused by distortion or loss of high frequency information in image.
- Ringing effect is usually introduced to image after different image processing algorithms. The most often it appears after image and video compression.

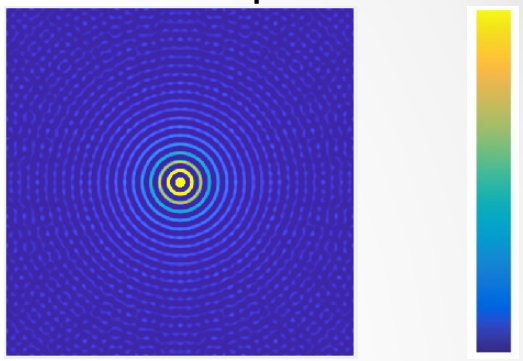


Ideal low pass filter

Ideal low pass filter in frequency domain



Ideal low pass filter in spatial domain



Original Lena

Idea low pass Lena

Gaussian low pass filter

Gaussian function in frequency domain is another Gaussian

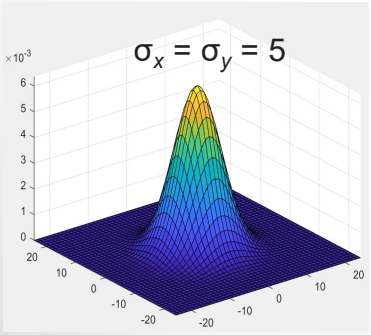
$$g(x) = Ae^{-\left(\frac{x^2}{2\sigma^2}\right)} \quad \text{where } A = \frac{1}{\sigma\sqrt{2\pi}}$$

$$G(\omega) = e^{-\left(\frac{\omega^2\sigma^2}{2}\right)}$$

σ is the standard deviation controlling the width of the curve

Idea low pass

Does Gaussian low pass filter suffer from the ringing effect?



Gaussian sigma=2

Gaussian sigma=3

Halftone

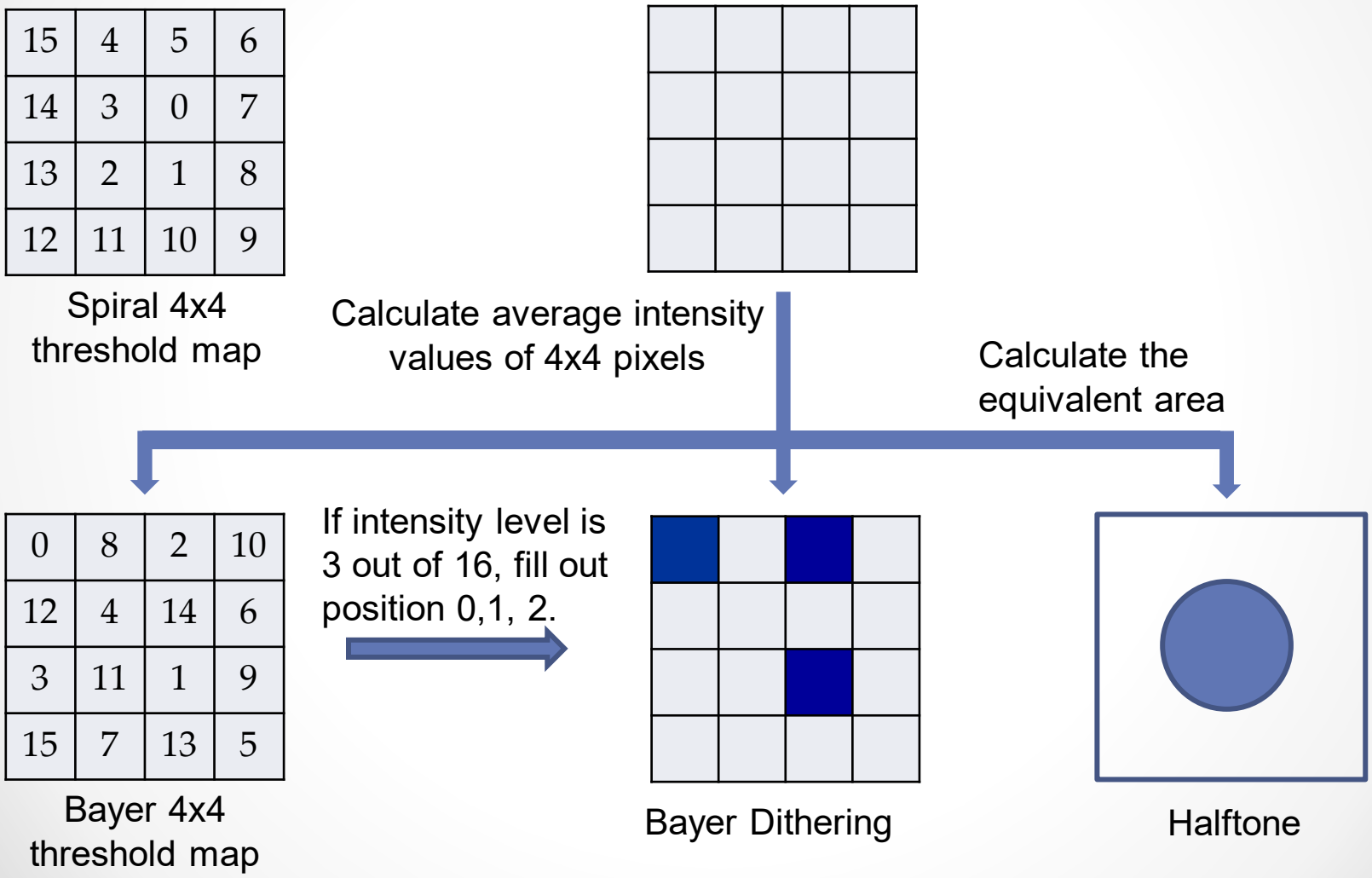
- Photographs use halftoning techniques to improve image quality. If you look closely at a gray image in a magazine, you will see the effect.
- Halftone is partial tone. When spray ink onto paper, it is either spray or not spray.
- Halftone simulates continuous tone imagery (e.g. gray scale or color images) through the dots, varying either in size or in spacing, thus generating a gradually changing intensity effect. Human eye would see this effect from a sufficient distance.
- E.g. for every 4x4 pixels in an image, calculate the average intensity of the 4x4 pixel. The 4x4 pixels are replaced by one black circle in the halftone image. An 256x256 image replaced by 64x64 circles in halftone image.

Halftone

Halftone

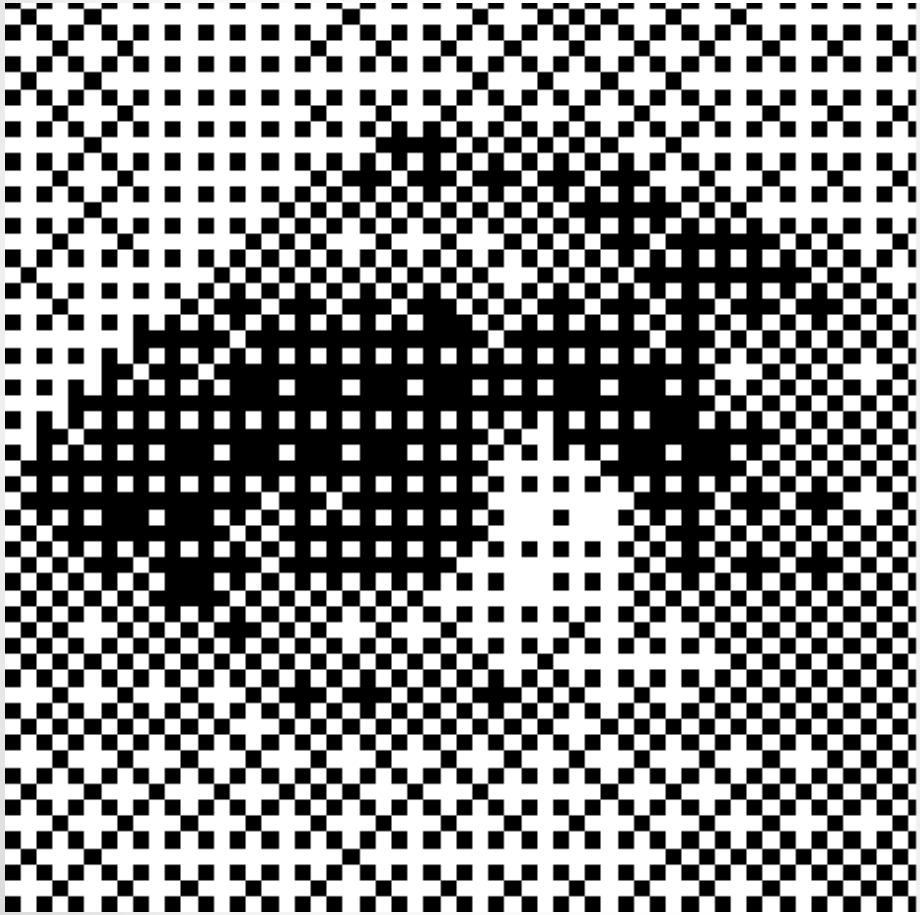
Cannot see? Must see from a sufficient distance!

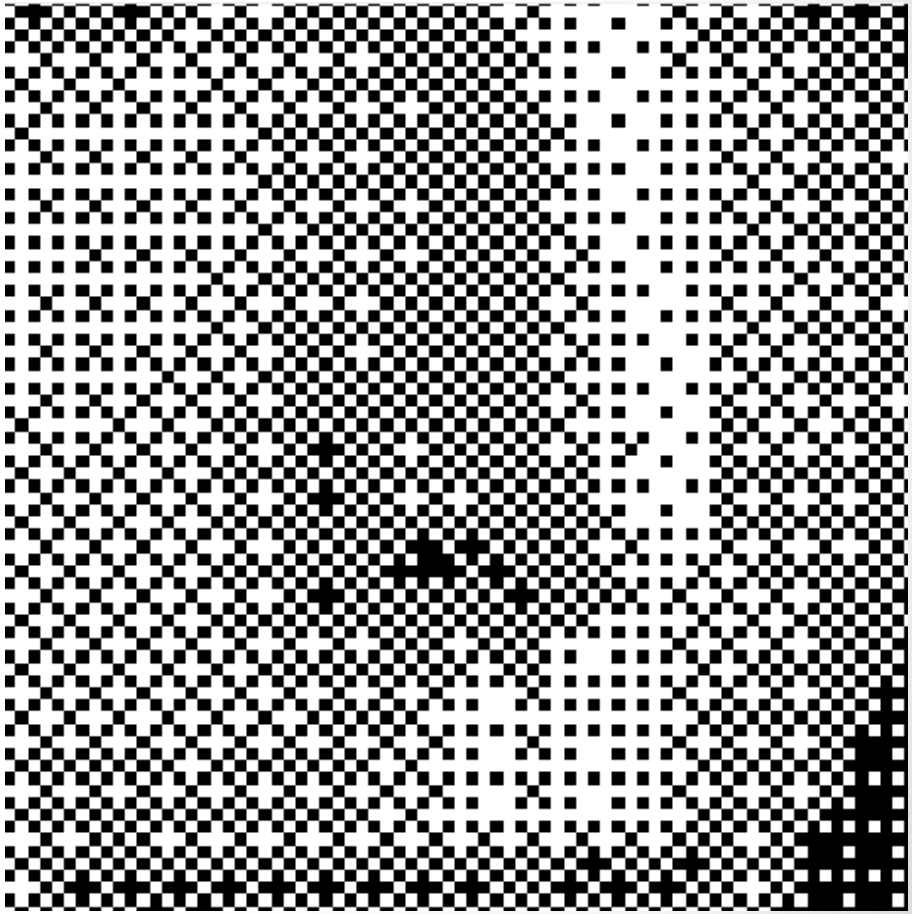
Halftone and Bayer Dithering



Compare Halftone and Bayer dithering

Bayer dithering





Color halftone

Cyan C Magenta M Yellow Y Black K

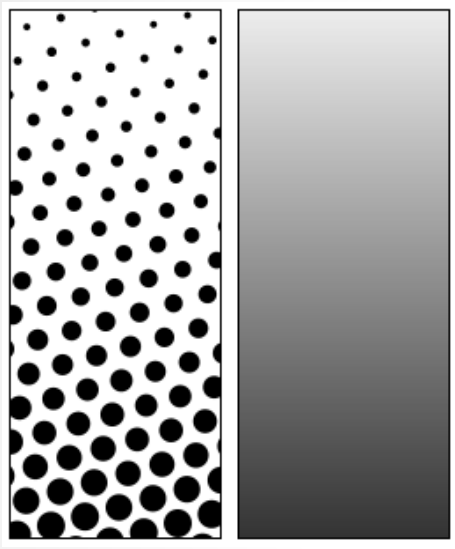


Image is obtained from https://en.wikipedia.org/wiki/Halftone#/media/File:Halftoning_introduction.svg

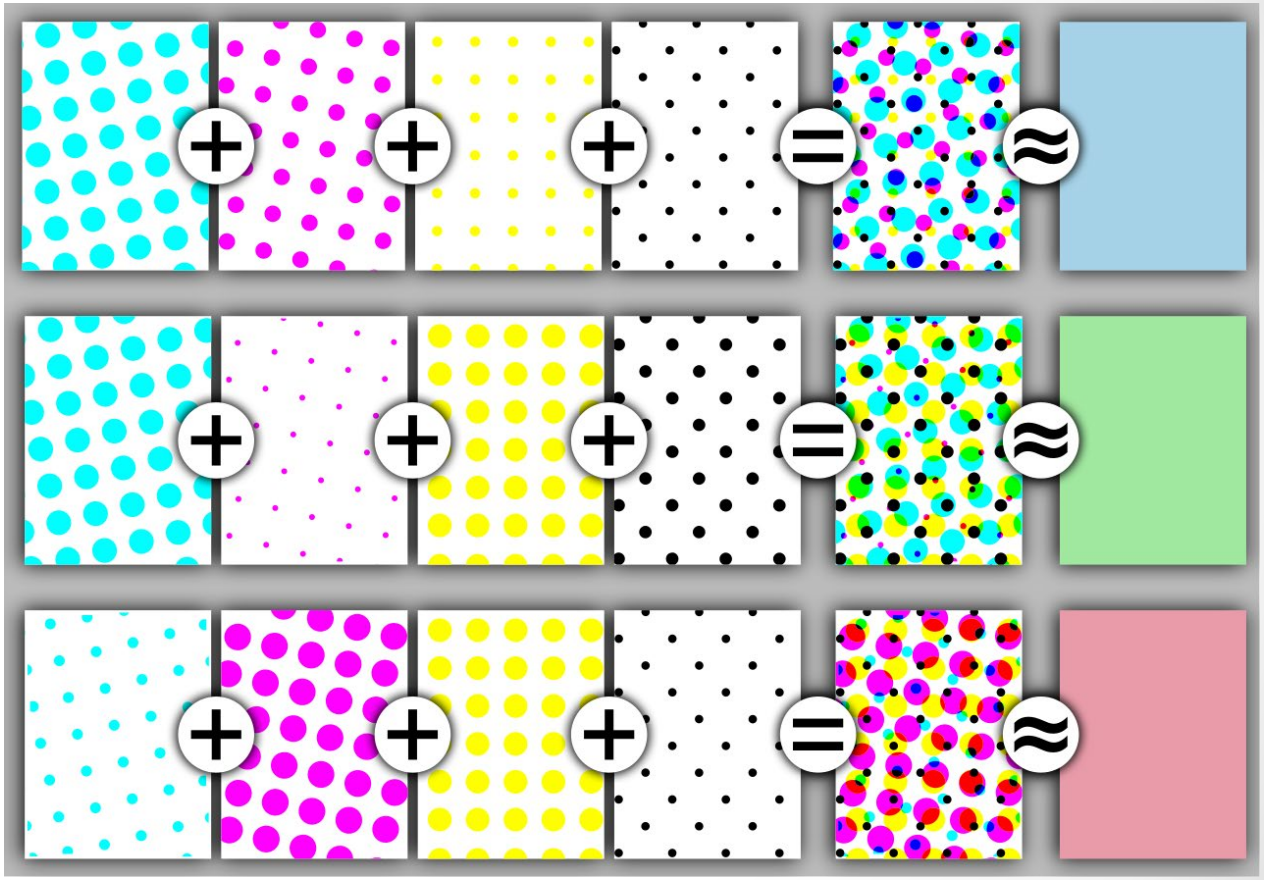
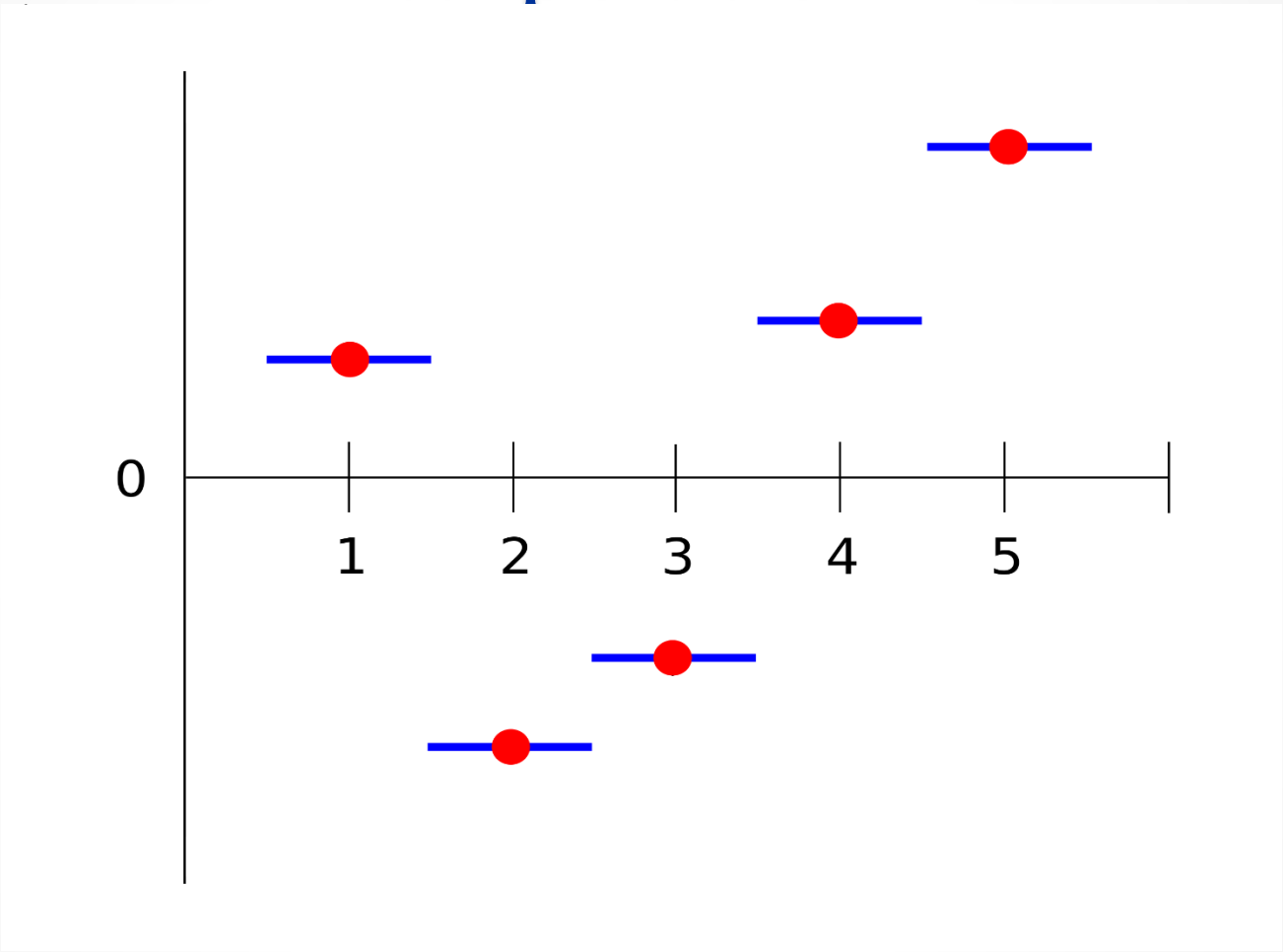


Image is obtained from https://en.wikipedia.org/wiki/Halftone#/media/File:Halftoningcolor.svg

Interpolation techniques

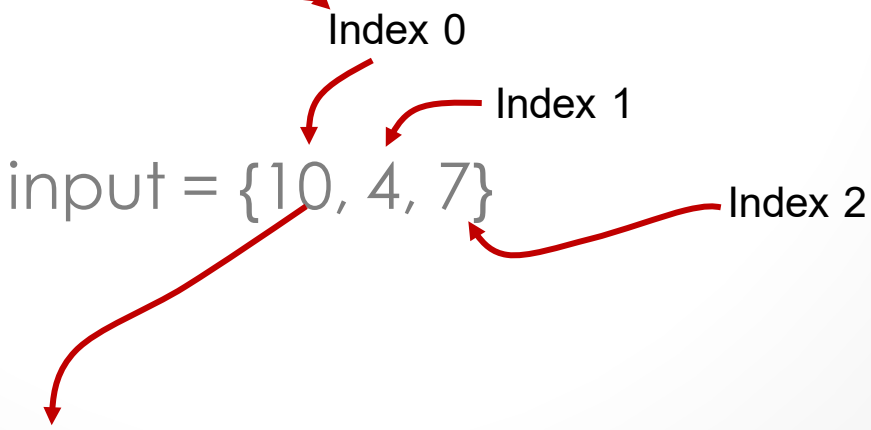
- 1D interpolation
  - zero-order (nearest-neighbor)
  - first-order (linear)
  - third-order (cubic)
- 2D interpolation: first interpolate the columns and then interpolate the resulting rows. Row-column or column-row doesn't matter.

Zero-order (nearest-neighbor) interpolation



Zero-order (nearest-neighbor) interpolation

- assume 3 to 5 interpolation
- => ratio = 5/3, n=5
- index = floor$\left(\frac{X-0.5}{ratio}\right)$ $\{X: 1, \ldots., n\}$
- index = floor{0.3, 0.9, 1.5, 2.1, 2.7}
  = {0, 0, 1, 2, 2}



- Output = {10, 10, 4, 7, 7}

First-order (bilinear) interpolation

- Bilinear interpolation is 2D version of using 1D linear interpolation

- $\frac{y-y_0}{x-x_0} = \frac{y_1-y_0}{x_1-x_0}$

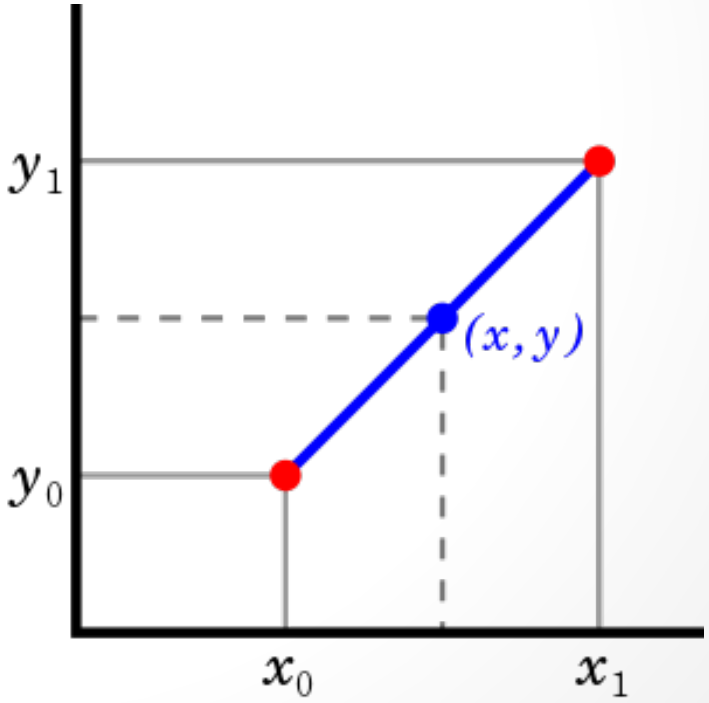


Figure obtained from Wikipedia

Third-order (bicubic) interpolation

- Need 4 points to compute the third order polynomial.
- $Y_x = a_0 + a_1 x + a_2 x^2 + a_3 x^3$
  - $Y_{-1} = a_0 - a_1 + a_2 - a_3$
  - $Y_0 = a_0$
  - $Y_1 = a_0 + a_1 + a_2 + a_3$
  - $Y_2 = a_0 + 2a_1 + 4a_2 + 8a_3$
- *Solve the above equations*
  - $a_0 = Y_0$
  - $a_1 = -\frac{1}{3}Y_{-1} - \frac{1}{2}Y_0 + Y_1 - \frac{1}{6}Y_2$
  - $a_2 = \frac{1}{2}Y_{-1} - Y_0 + \frac{1}{2}Y_1$
  - $a_3 = -\frac{1}{6}Y_{-1} + \frac{1}{2}Y_0 - \frac{1}{2}Y_1 + \frac{1}{6}Y_2$

Interpolation methods

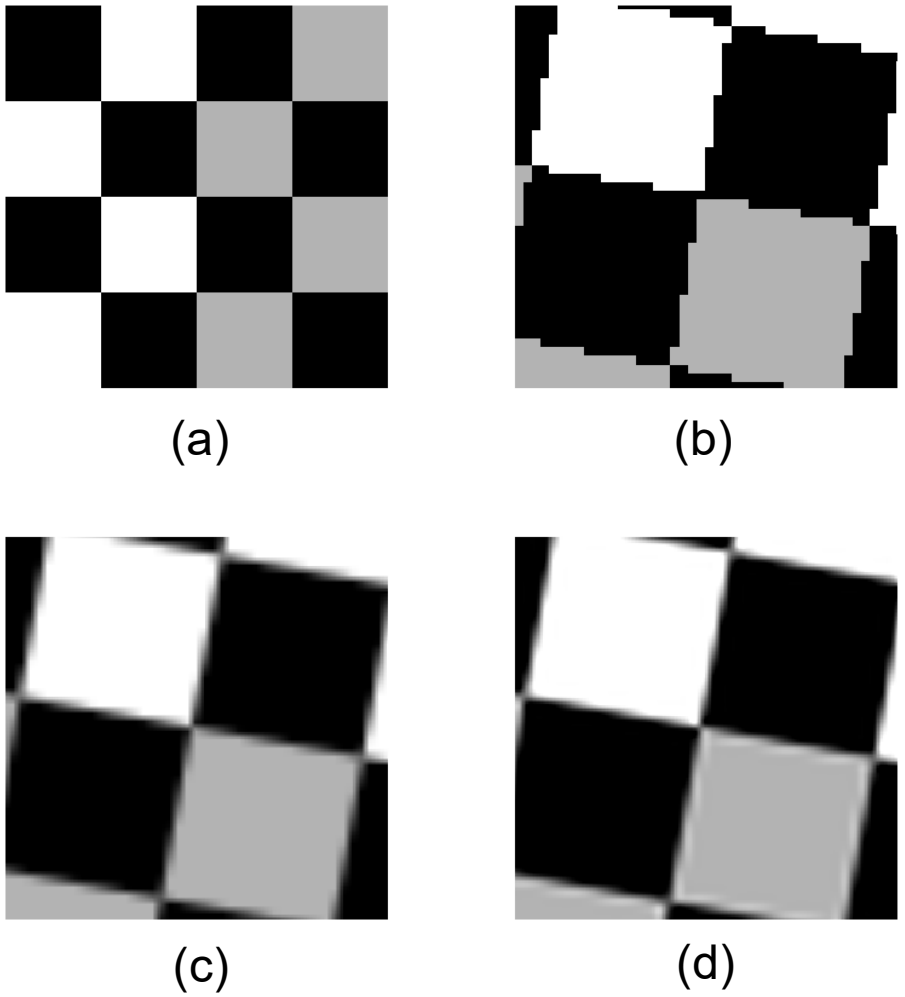
Effects of different interpolation techniques on 5x interpolation, zoom-in and rotated 100◦ clockwise of original images:

(a) original image;

(b) zero-order (nearest-neighbor) interpolation;

(c) first-order (bilinear) interpolation;

(d) third-order (bicubic) interpolation.



Interpolation methods

Effects of different interpolation techniques on 5x interpolation of original Lena images:
(a) original image;
(b) zero-order (nearest-neighbor) interpolation;
(c) first-order (bilinear) interpolation;
(d) third-order (bicubic) interpolation.

(a)

(b)

(c)

(d)