IMAGE BLURRING

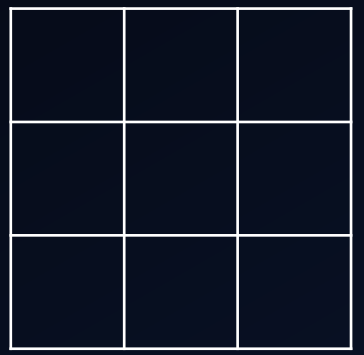
- Convolutions and Kernels
- Median filter
- Box blur
- Gaussian blur

Author: prof. Yevhenii Borodavka

Convolutions and Kernels

The small group of pixels is a small 3x3 or a 5x5 image called a **Kernel** represented by a simple matrix. Think of this approach as using a very small image template that is 3x3 or 5x5 pixels in size as opposed to a larger image template.

We are going to dig into a technique that is incredibly important in computer vision. This fundamental technique is known as convolution. A convolution is done by multiplying a pixel and its neighboring pixel's color values by a kernel using a sliding window. Convolutions and Kernels manipulate pixels not solely based on the value of the pixel itself, but on the pixels in the immediate vicinity of a particular pixel (sometimes referred to as Connected Pixels or Neighbors). A convolution is just another type of Image.



Kernel 3x3 Pixels

Median filter

Median filtering is a nonlinear method used to remove noise from images. It is widely used as it is very effective at removing noise while preserving edges. It is particularly effective at removing ‘salt and pepper’ type noise. The median filter moves through the image pixel by pixel, replacing each value with the median value of neighboring pixels. The pattern of neighbors is called the "window", which slides, pixel by pixel, over the entire image. The median is calculated by first sorting all the pixel values from the window into numerical order, and then replacing the pixel being considered with the middle (median) pixel value.

Use of a median filter to improve an image severely corrupted by defective pixels.

Median filter

The following example shows the application of a median filter to a simple one-dimensional signal. A window size of three is used, with one entry immediately preceding and following each entry. For the edges, we do padding — repeat the border pixels.

X =	3	9	4	52	3	8	6	2	2	9

Y[0] = median[3 3 9] = 3
Y[1] = median[3 4 9] = 4
Y[2] = median[4 9 52] = 9
Y[3] = median[3 4 52] = 4
Y[4] = median[3 8 52] = 3
Y[5] = median[3 6 8] = 6
Y[6] = median[2 6 8] = 6
Y[7] = median[2 2 6] = 2
Y[8] = median[2 2 9] = 2
Y[9] = median[2 9 9] = 9

Y =	3	4	9	4	8	6	6	2	2	9

Median filter

The 2D median filter example with a 3x3 window with extended borders.

1	4	0	1	3	1
2	2	4	2	2	3
1	0	1	0	1	0
1	2	1	0	2	2
2	5	3	1	2	5	5
1	1	4	2	3	0	0
				3	0	0

Input

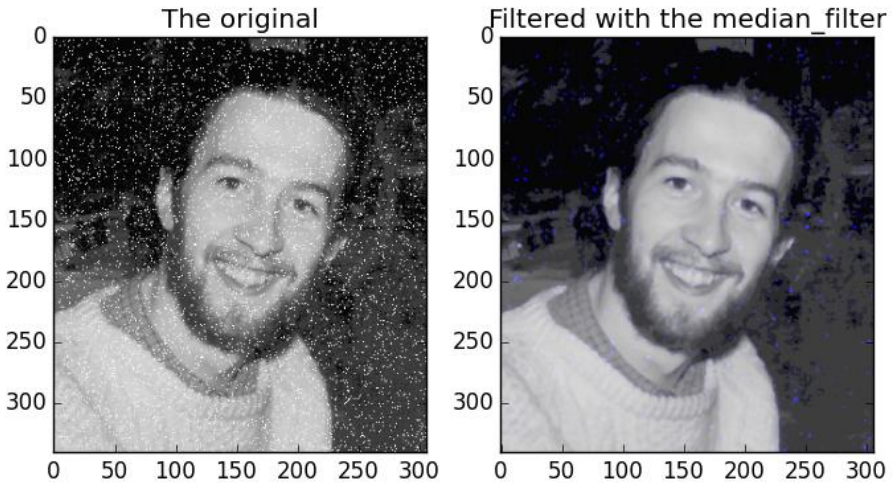
Sorted [0 0 1 1 1 2 2 4 4] = 1
Sorted [0 0 0 1 1 2 2 4 4] = 1
Sorted [0 0 0 0 2 3 3 5 5] = 2

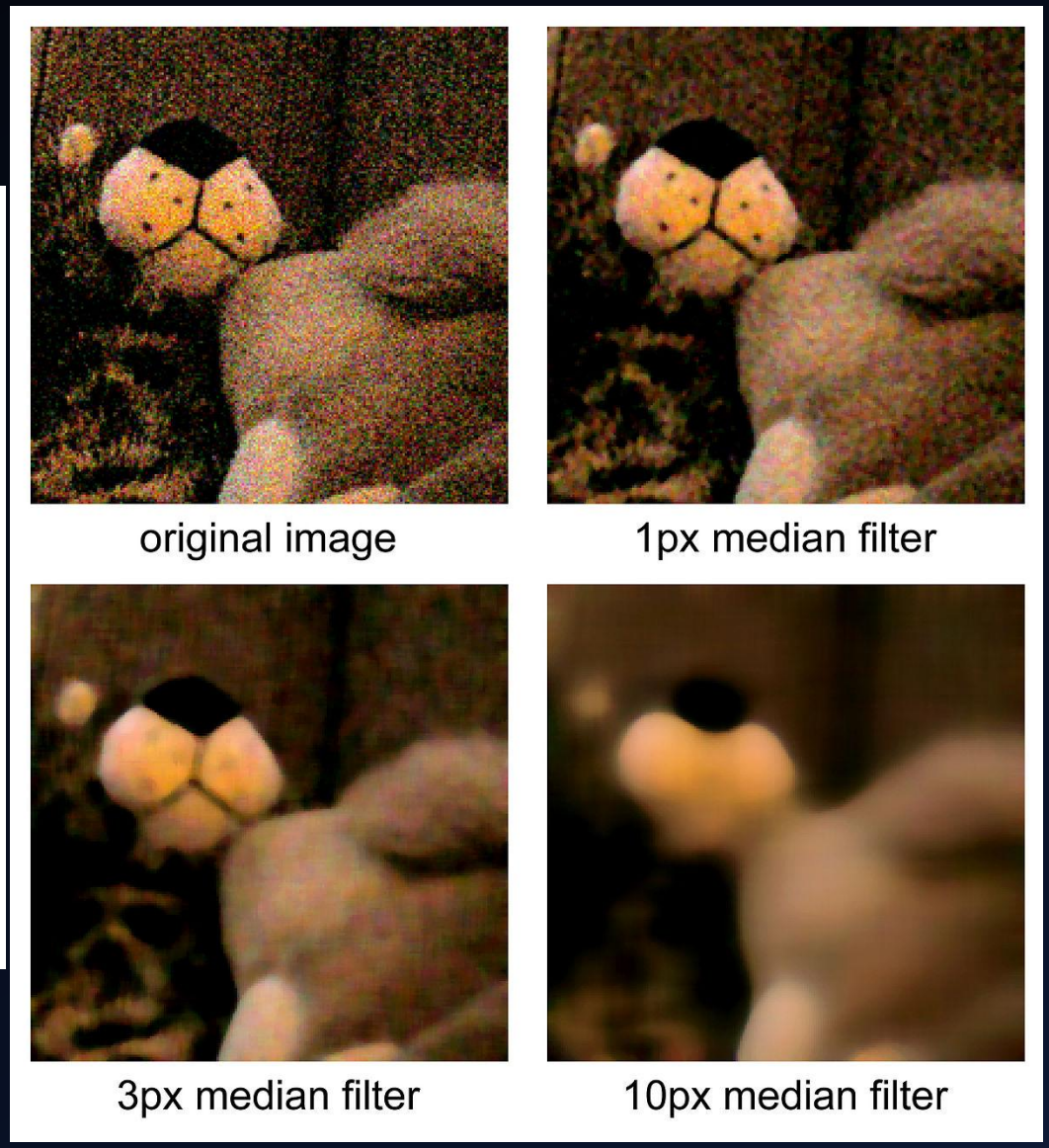
2	2	2	2	2	2
1	1	1	1	1	1
1	1	1	1	2	2
1	1	1	1	1	2
1	2	2	2	2	2
1	2	2	3	2	2

Output

Median filter

The Median filter examples





Box blur

The Box Blur (sometimes referred to as Mean or Average Blur), looks at each pixel of an image and replaces its value with the average value of all of its surrounding pixels. In our example, we will be using a simple 3x3 matrix kernel where all values are 1. The box blur kernel:

1	1	1
1	1	1
1	1	1

While the above kernel is incredibly simple — it's going to help us understand how convolutions work. In this convolution, because the matrix values are all 1, the result of each window is the average value of the center pixel and its neighbors. Since we are using a sliding window, we cannot update the pixel value in-line (otherwise it would impact the next time the window shifts). We update a new image with the average value at the position corresponding to the center of the kernel.

Box blur

The Box Blur is an example of a convolution. It uses a sliding window the size of the kernel to calculate the average of a neighborhood of pixels.

1. Multiply the 1st neighbor pixel by the 1st value in the Kernel

125	213	98	203	202	170
104	145	161	204	201	157
72	8	209	202	194	144
73	9	202	201	194	156
81	15	189	185	181	144
15	189	185	194	227	158

Original Image

1	1	1
1	1	1
1	1	1

3x3 Box
Blur Kernel

Box blur

Using a sliding window, the convolution will process all of the pixels in the image. Once the convolution is completed, our new image will appear blurred.

Box Blur Convolution

125	213	98	203	202	170
104	145	161	204	201	157
72	8	209	202	194	144
73	9	202	201	194	156
81	15	189	185	181	144
15	189	185	194	227	158

Original Image

Blurred Image

Box blur

In the above animation, notice the pixels with the values of 8 and 9. These two pixels, in particular, are outliers compared to all the other pixels in the images. These pixels stand out from the rest of the pixels. After the convolution, these pixel values are replaced with 109 and 95 bringing those values more in alignment with the neighbors.

Also visible in the animation is that the pixels closest to the edges are not calculated in the convolution. There are methods of dealing with those border pixels in three different ways. The pixels can be discarded, they can be carried over from the original, or an average of the available pixels can be calculated (padding or mirroring).

Let’s consider the image 6x6 pixels and kernel 5x5 pixels. The image should be extended with 2 pixels in each direction.

Box blur

1	1	1	4	0	1	3	1	1	1
1	1	1	4	0	1	3	1	1	1
1	1	**1**	**4**	**0**	**1**	**3**	1	1	1
2	2	**2**	**2**	**4**	**2**	**2**	3	3	3
1	1	**1**	**0**	**1**	**0**	**1**	0	0	0
1	1	**1**	**2**	**1**	**0**	**2**	2	2	2
2	2	**2**	**5**	**3**	**1**	**2**	5	5	5
1	1	**1**	**1**	**4**	**2**	**3**	0	0	0
1	1	1	1	4	2	3	0	0	0
1	1	1	1	4	2	3	0	0	0

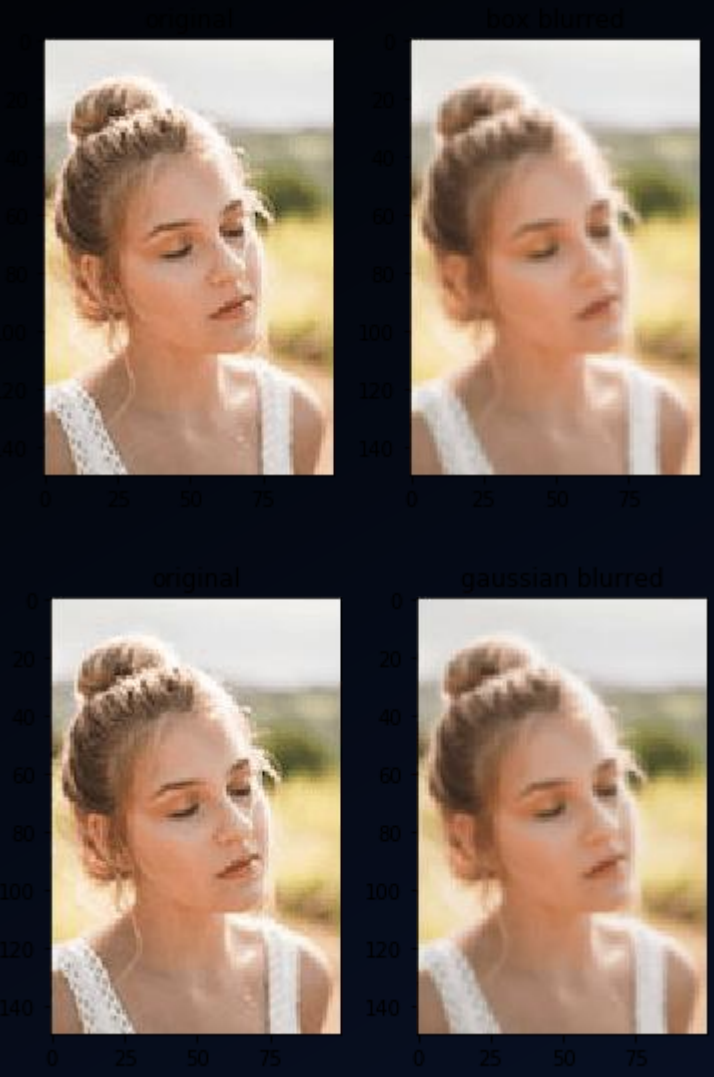
Padding

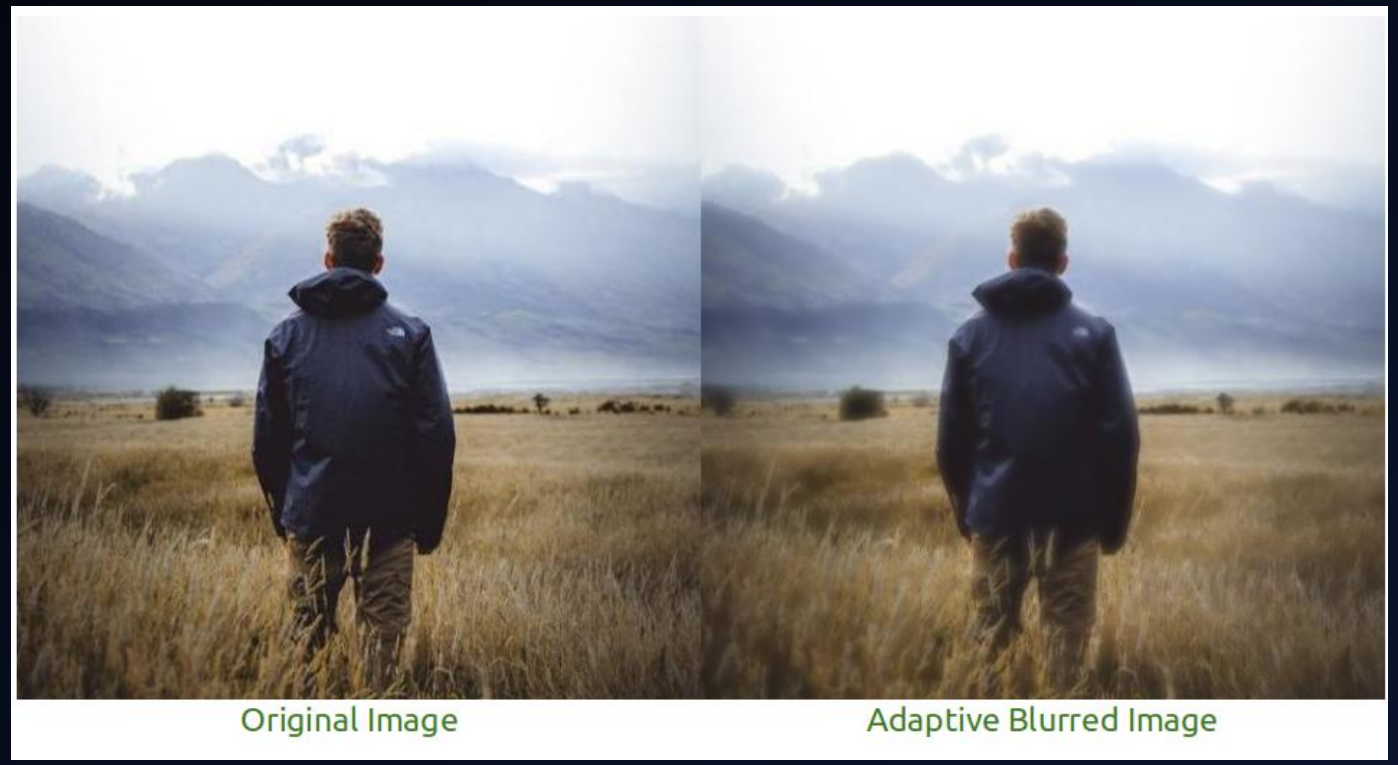
2	2	2	2	4	2	2	3	3	2
4	1	1	4	0	1	3	1	1	3
4	1	**1**	**4**	**0**	**1**	**3**	**1**	1	3
2	2	**2**	**2**	**4**	**2**	**2**	**3**	3	2
0	1	**1**	**0**	**1**	**0**	**1**	**0**	0	1
2	1	**1**	**2**	**1**	**0**	**2**	**2**	2	2
5	2	**2**	**5**	**3**	**1**	**2**	**5**	5	2
1	1	**1**	**1**	**4**	**2**	**3**	**0**	0	3
1	1	1	1	4	2	3	0	0	3
5	2	2	5	3	1	2	5	5	2

Mirroring

Box blur

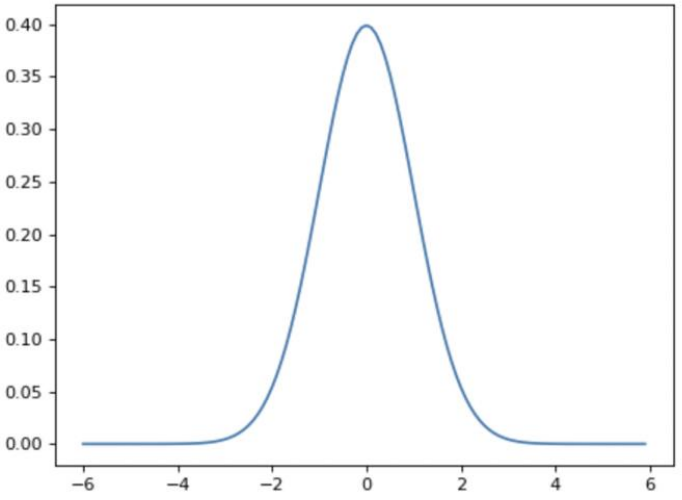
The Box Blur examples

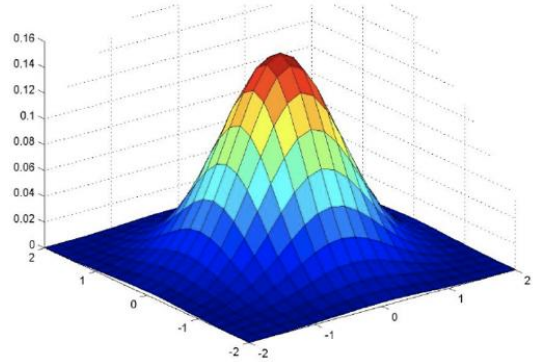




Gaussian blur

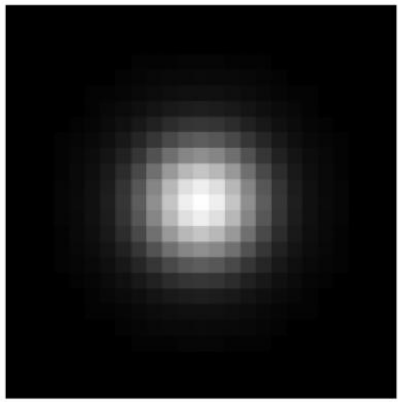
Gaussian kernel, as its name implies, has the shape of the function ‘Gaussian distribution’ to define the weights inside the kernel, which are used to compute the weighted average of the neighboring points (pixels) in an image.





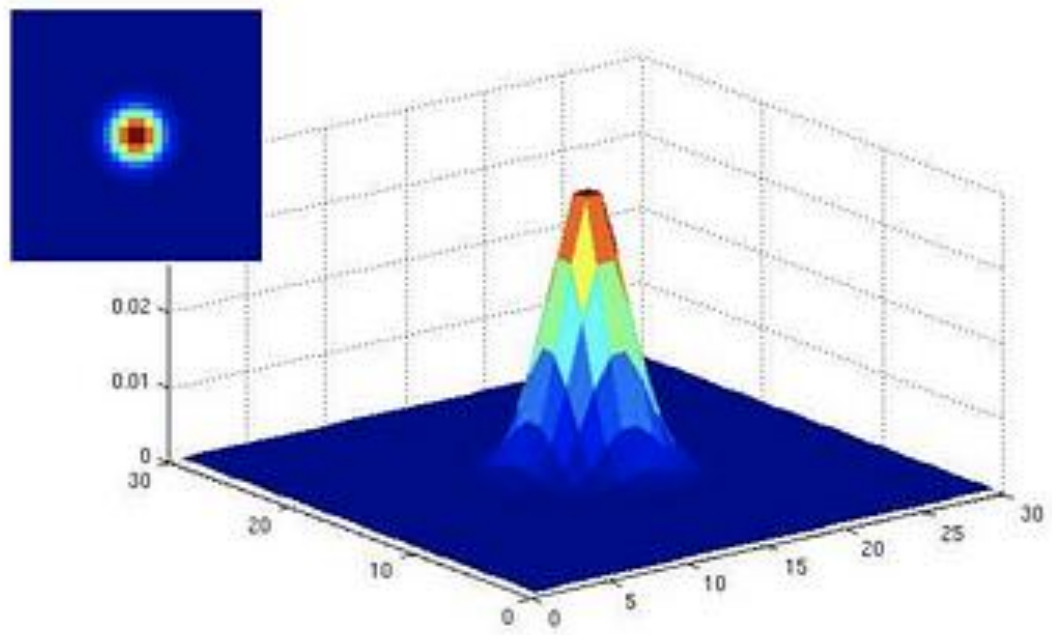
Gaussian kernel

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

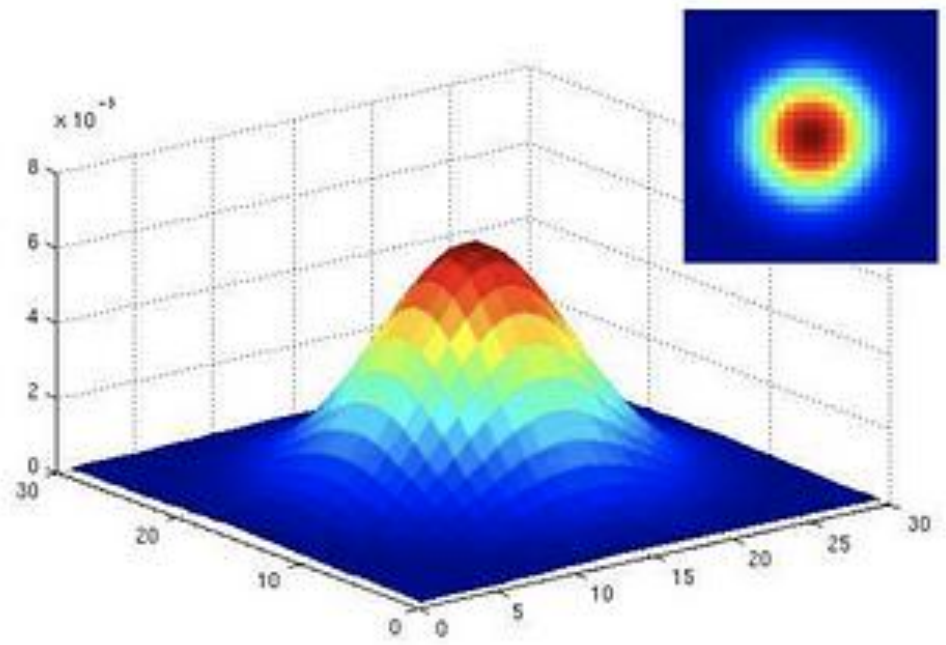


Gaussian blur

In other words, each value in the Gaussian filter is from the zero mean Gaussian distribution. One thing we need to keep in mind is that the kernel size is dependent on the standard deviation σ of the Gaussian function:



$\sigma = 2$ with 30×30 kernel



$\sigma = 5$ with 30×30 kernel

Gaussian blur

By setting the standard deviation σ, we can control to what certain extent we smooth the image. In other words, the higher the standard deviation gets the stronger effect of smoothing effect the image has.
Here is one example of a simple and easy-to-read 3x3 Gaussian Kernel:

1	2	1
2	4	2
1	2	1

The convolution of a Gaussian kernel is identical to the Box Blur kernel. Think of the box blur as dividing the sum of the kernel values, which totals 9. In the Gaussian kernel illustrated above, the sum of the kernel values is 16.

Gaussian blur

The Gaussian convolution is identical to the Box Blur — we just use a different set of kernel values

1. Multiply the 1st neighbor pixel by the 1st value in the Kernel

125	213	98	203	202	170
104	145	161	204	201	157
72	8	209	202	194	144
73	9	202	201	194	156
81	15	189	185	181	144
15	189	185	194	227	158

Original Image

1	2	1
2	4	2
1	2	1

3x3
Gaussian Kernel

Gaussian blur

The Gaussian Kernels with sizes 3x3, 5x5 and 7x7 for $\sigma \sim 1$:

1/16

1	2	1
2	4	2
1	2	1

1/273

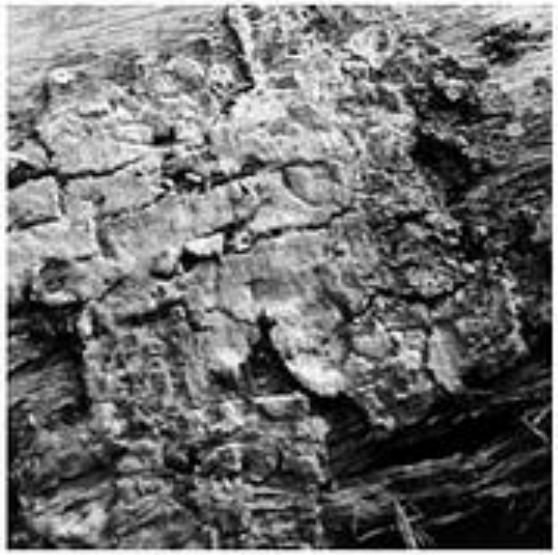
1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

1/1003

0	0	1	2	1	0	0
0	3	13	22	13	3	0
1	13	59	97	59	13	1
2	22	97	159	97	22	2
1	13	59	97	59	13	1
0	3	13	22	13	3	0
0	0	1	2	1	0	0

Gaussian blur

For comparison, here are the results of both a Box and Gaussian Blur



Original Image

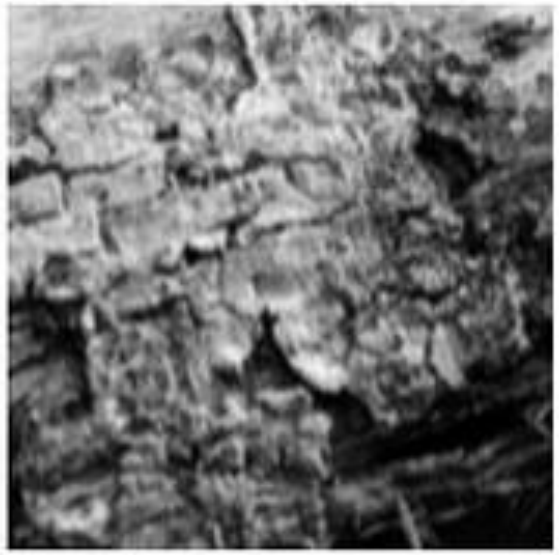


Image with Box Blur

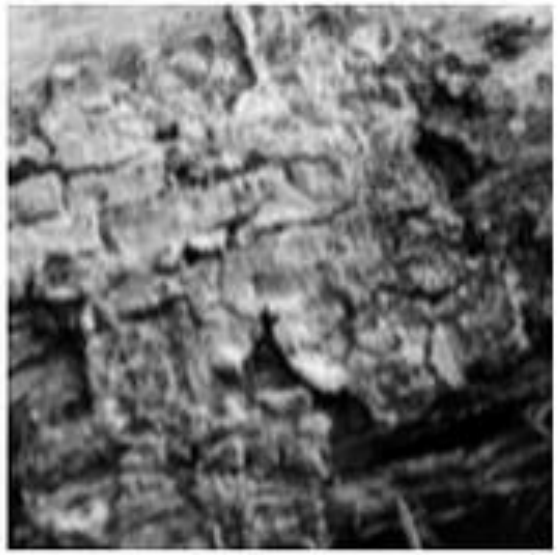
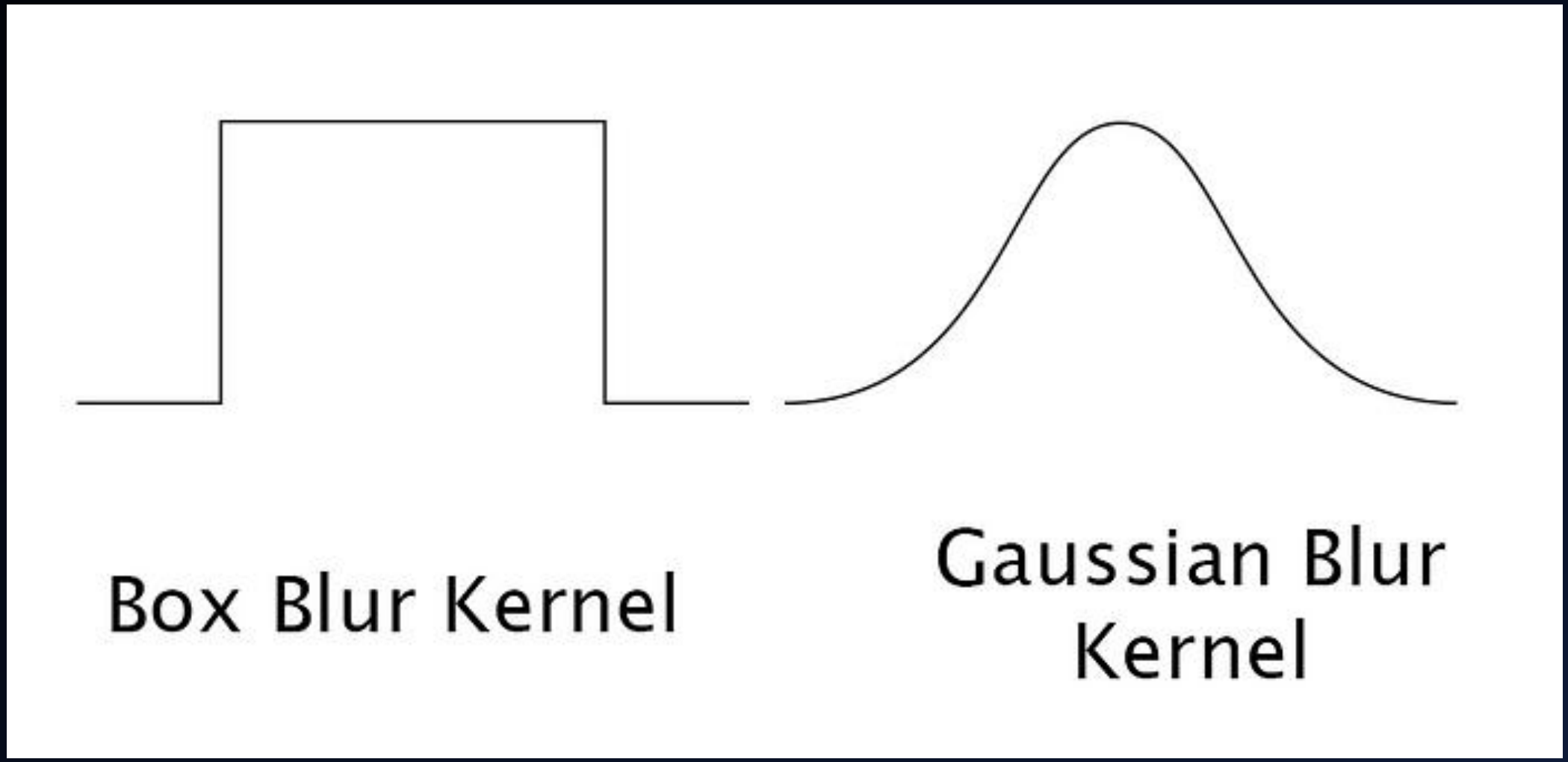


Image with Gaussian Blur

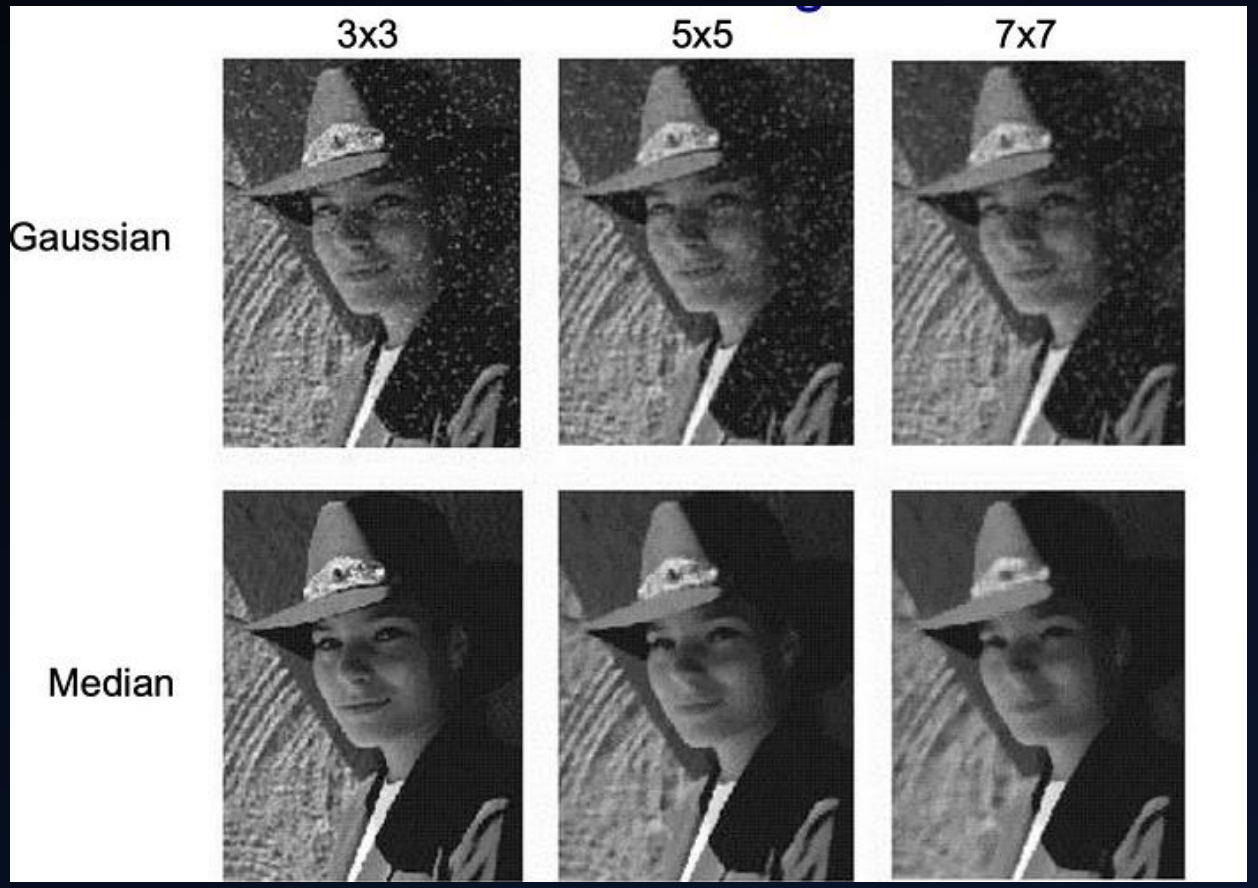
Gaussian blur

As a quick visualization exercise, let's also compare what the Box Blur and Gaussian Blur kernels would look like if we were to plot them in a 2D graph.



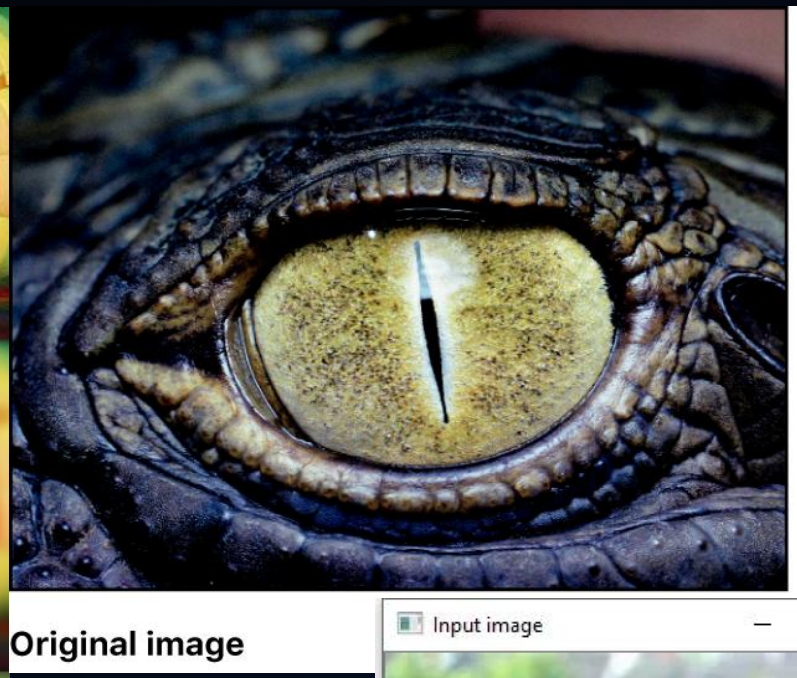
Gaussian blur

Here is a visual comparison between Gaussian and Median filter. We can observe that when the noise level is too high, although the amount of noise pixels decreases with increasing Gaussian filter size, they still exist in the image. The median filter, on the other hand, already removes most of the noise pixels with a 3x3 filter size.

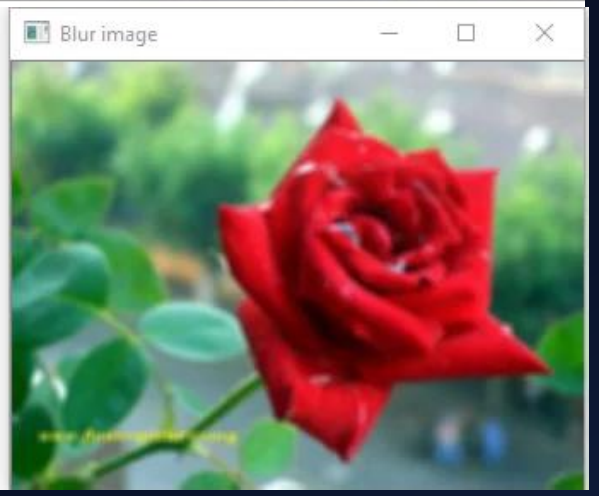


Gaussian blur

The Gaussian Blur examples



Original image



Thank you!