Convolution

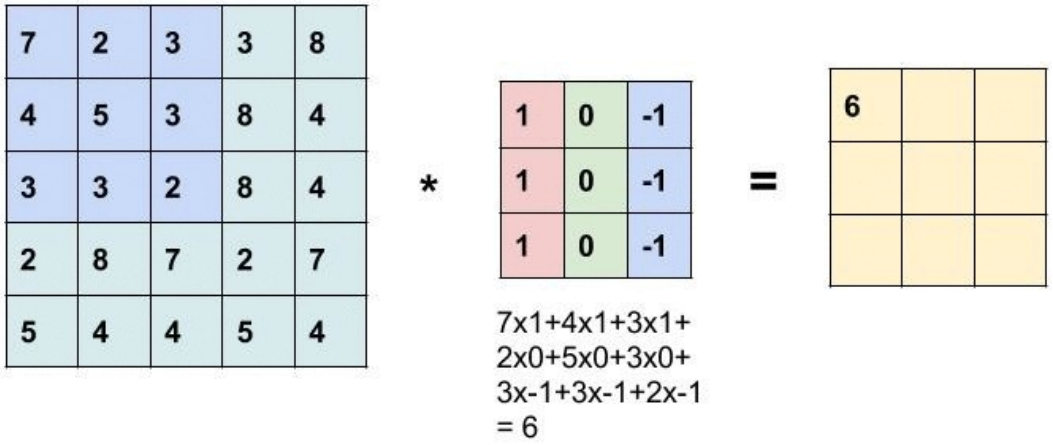


Image Smoothing - Mean filter

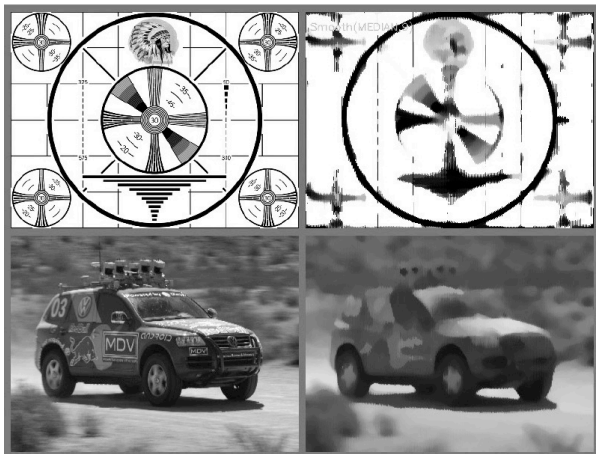
```
void blur(    InputArray src,
              OutputArray dst,
              Size ksize,
              Point anchor = Point(-1,-1),
              int borderType = BORDER_DEFAULT
)
```

Example code:
blur(image, result, Size(3, 3));

Detailed textures are lost.

Image Smoothing - Median filter

```
void medianBlur(    InputArray src,
                    OutputArray dst,
                    int ksize
)
```



Example code:
medianBlur(image, result, 5);

Many thin textures and lines are lost.
There are several areas of semi-equal color.

Image Smoothing - Median filter

Median filter is particularly useful to combat salt-and-pepper noise.

Compare with Mean filter.

salted.bmp

Image Smoothing - Gaussian filter

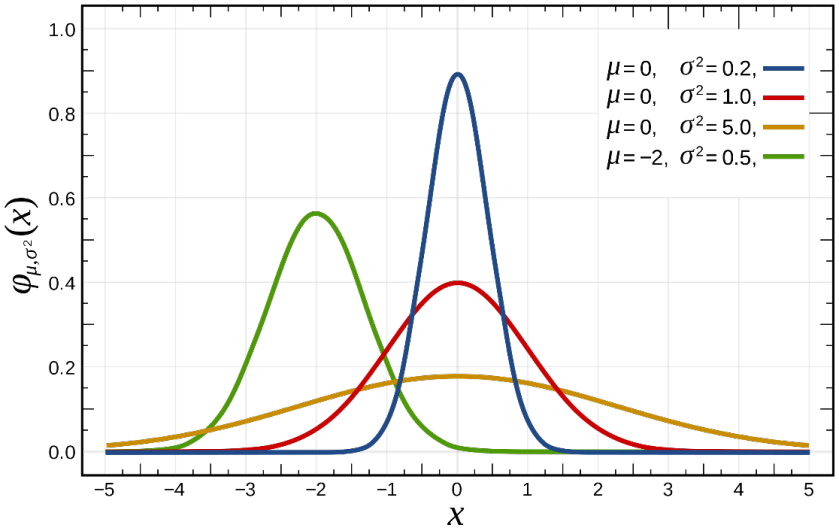
```
void GaussianBlur(      InputArray src,
                        OutputArray dst,
                        Size ksize,
                        double sigmaX,
                        double sigmaY = 0,
                        int borderType = BORDER_DEFAULT
)
```

Example code:
GaussianBlur(image, result, Size(3, 3), 0, 0);

Gaussian smoothing reduces noise while preserving signal.
Although the image is blurred, the textures are not lost.

Gaussian function

$$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



Gaussian function

$$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

$$\begin{bmatrix} 0.045 & 0.122 & 0.045 \\ 0.122 & 0.332 & 0.122 \\ 0.045 & 0.122 & 0.045 \end{bmatrix}$$

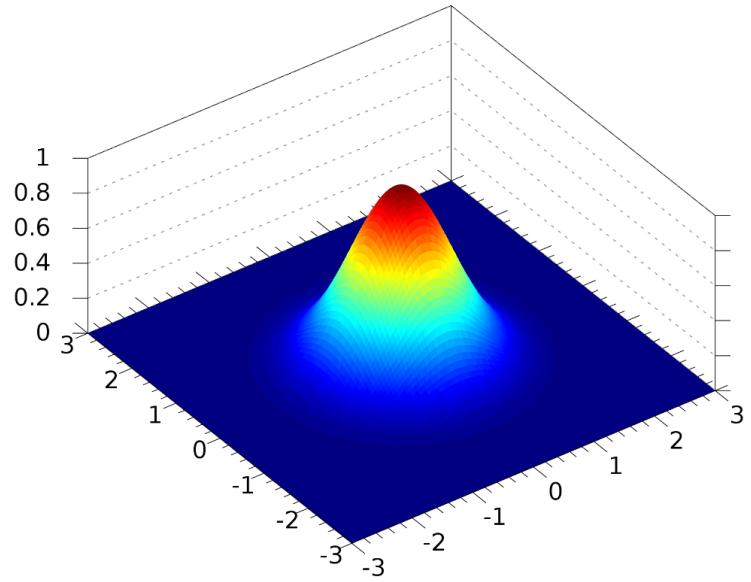
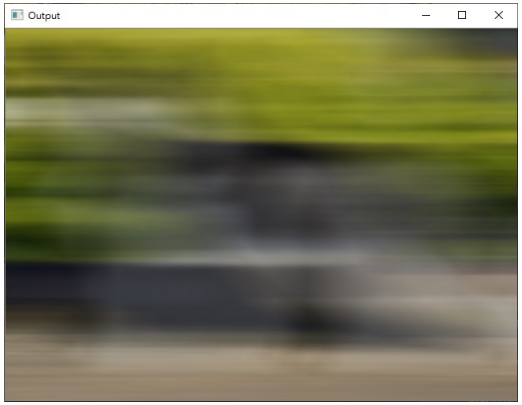


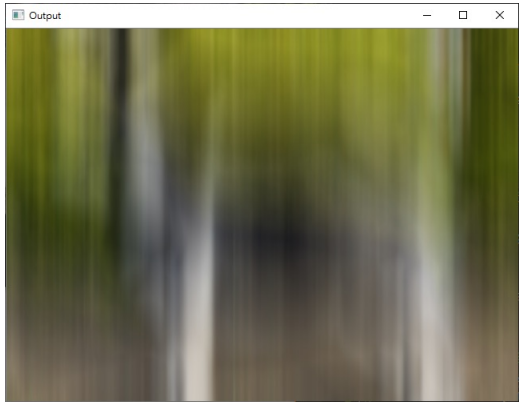
Image Smoothing - Gaussian filter

- Try to use different kernel size and sigma value
- ksize.width and ksize.height can differ but they both must be positive and odd. Or, they can be zero's and then they are computed from sigma.

Example code:

```
GaussianBlur(image, result, Size(0, 0), 100, 1);
GaussianBlur(image, result, Size(0, 0), 1, 100);
```





Practice

- Write a program to blur an image
- Do not use built in functions
- Use a 3 * 3 filter