In this course, we will mostly restrict our concept of images to **grayscale**. Grayscale values usually have a resolution of 8 bits (256 different values), in medical applications sometimes 12 bits (4096 values), or in binary images only 1 bit (2 values). We simply choose the available gray level whose intensity is closest to the gray value color we want to convert.

Digitizing Visual Scenes

With regard to spatial resolution, we will map the intensity in our image onto a two-dimensional finite array:

```
[0, 0] [0, 1] [0, 2] [0, 3]
[1, 0] [1, 1] [1, 2] [1, 3]
[2, 0] [2, 1] [2, 2] [2, 3]
```

So the result of our digitization is a two-dimensional array of discrete intensity values. Notice that in such a digitized image \( F[i, j] \)
- the **first coordinate** \( i \) indicates the row of a pixel, starting with 0,
- the **second coordinate** \( j \) indicates the column of a pixel, starting with 0.

In an \( m \times n \) pixel array, the relationship between image (with origin in int center) and pixel coordinates is given by the equations

\[
x' = j - \frac{n-1}{2} \quad y' = -\left( i - \frac{m-1}{2} \right)
\]
Intensity Transformation

Sometimes we need to transform the intensities of all image pixels to prepare the image for better visibility of information or for algorithmic processing.

Gamma Transformation

Gamma transformation:

Linear Histogram Scaling

For a desired intensity range \([a, b]\) we can use the following linear transformation:

\[
I' = \frac{b - a}{I_{\text{max}} - I_{\text{min}}} (I - I_{\text{min}}) + a
\]

Note that outliers (individual pixels of very low or high intensity) should be disregarded when computing \(I_{\text{min}}\) and \(I_{\text{max}}\).

Image Filtering

- Many basic image processing techniques are based on convolution.
- In a convolution, a convolution filter \(W\) is applied to every pixel of an image \(I\) to create a filtered image \(I'\).
- The filter \(W\) itself is a 2D matrix of real values.
- To simplify the mathematics, we could consider \(W\) to have a center \([0, 0]\) and extend from \(-m\) to \(m\) vertically and \(-n\) to \(n\) horizontally.
- This means that \(W\) is of size \((2m + 1) \times (2n + 1)\).
Convolution

Example: Averaging filter:

\[
\begin{array}{ccc}
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
\end{array}
\]

Grayscale Image:

\[
\begin{array}{cccc}
16 & 32 & 9 \\
5 & 1 & 0 \\
4 & 4 & 2 & 9 \\
\end{array}
\]

Averaging Filter:

\[
\begin{array}{ccc}
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
\end{array}
\]

Original Image:

\[
\begin{array}{cccc}
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
\end{array}
\]

Filtered Image:

\[
\begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 5 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{array}
\]


Now you can see the averaging (smoothing) effect of the 3x3 filter that we applied.

Convolution

- It needs to be noted that for convolution the filter needs to be rotated by 180° before starting the computations (otherwise it’s a correlation).
- An intuitive explanation is that we would like convolution to be like a “local multiplication of patterns.”
- For example, if our image contains a few 1s and otherwise 0s, we would expect the convolution result to contain a copy of the filter pattern centered at each 1.
- Let us look at an image with one 1-pixel:
Convolution

<table>
<thead>
<tr>
<th>Image:</th>
<th>Filter:</th>
<th>Result:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0</td>
<td>1 2 3 4 5</td>
<td>0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 0 0 0</td>
<td>4 5 6 7 8</td>
<td>0 9 8 7 0</td>
</tr>
<tr>
<td>0 0 1 0 0</td>
<td>7 8 9 0 0</td>
<td>0 6 5 4 0</td>
</tr>
<tr>
<td>0 0 0 0 0</td>
<td></td>
<td>0 3 2 1 0</td>
</tr>
<tr>
<td>0 0 0 0 0</td>
<td></td>
<td>0 0 0 0 0</td>
</tr>
</tbody>
</table>

Oops! The copy of the filter is rotated by 180°!

Convolution

• If we rotate the filter by 180° beforehand, we get the desired result.
• This leads to the following definition of convolution for image $I$, filter $W$, and result $I^*$:

$$I^*[i, j] = \sum_{k=-m}^{m} \sum_{l=-n}^{n} W[k, l] I[i-k, j-l]$$

• This formula needs to be applied to all coordinates $[i, j]$ in $I$ in order to create the complete image $I^*$.
• Convolution is commutative, i.e., $W$ and $I$ are exchangeable.

Image Filtering

More common: Gaussian Filters

Continuous version:

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

- implement decreasing influence by more distant pixels

Discrete version: $1/273$

<table>
<thead>
<tr>
<th>Original $3\times3$</th>
<th>9x9</th>
<th>$15\times15$</th>
</tr>
</thead>
</table>

Effect of Gaussian smoothing:

Different Types of Filters

• Smoothing can reduce noise in the image.
• This can be useful, for example, if you want to find regions of similar color or texture in an image.

• However, there are different types of noise.
• For so-called “salt-and-pepper” noise, for example, a median filter can be more effective.
• Note that it is not a convolution filter, but it works similarly.

Median Filter

• Use, for example, a 3x3 filter and move it across the image like we did before.
• For each position, compute the median of the brightness values of the nine pixels in question.
  – To compute the median, sort the nine values in ascending order.
  – The value in the center of the list (here, the fifth value) is the median.
• Use the median as the new value for the center pixel.
Median Filter

- **Advantage** of the median filter: Capable of eliminating outliers such as the extreme brightness values in salt-and-pepper noise.

- **Disadvantage**: The median filter may change the contours of objects in the image.