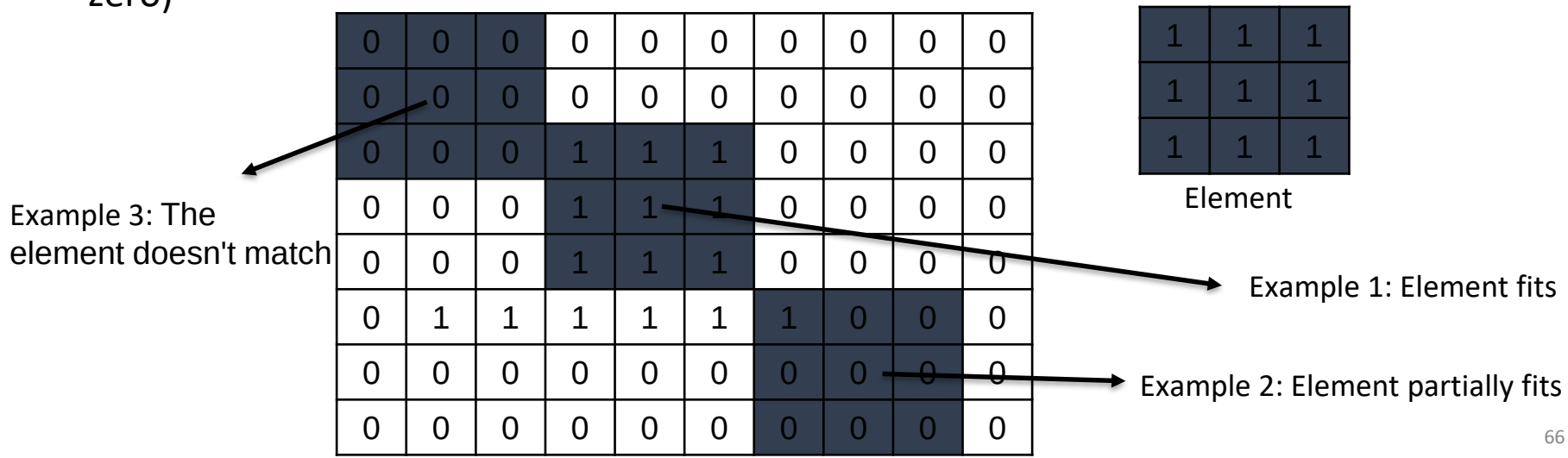
Morphological Operation

- Simple operations that are very often used on binary images
- To perform morphological operations, we need input image and kernel (structuring element)
- The morphological operations (most common) that we will discuss deeper are:
  - Erosion
  - Dilation
  - Opening
  - Closing
  - Gradient

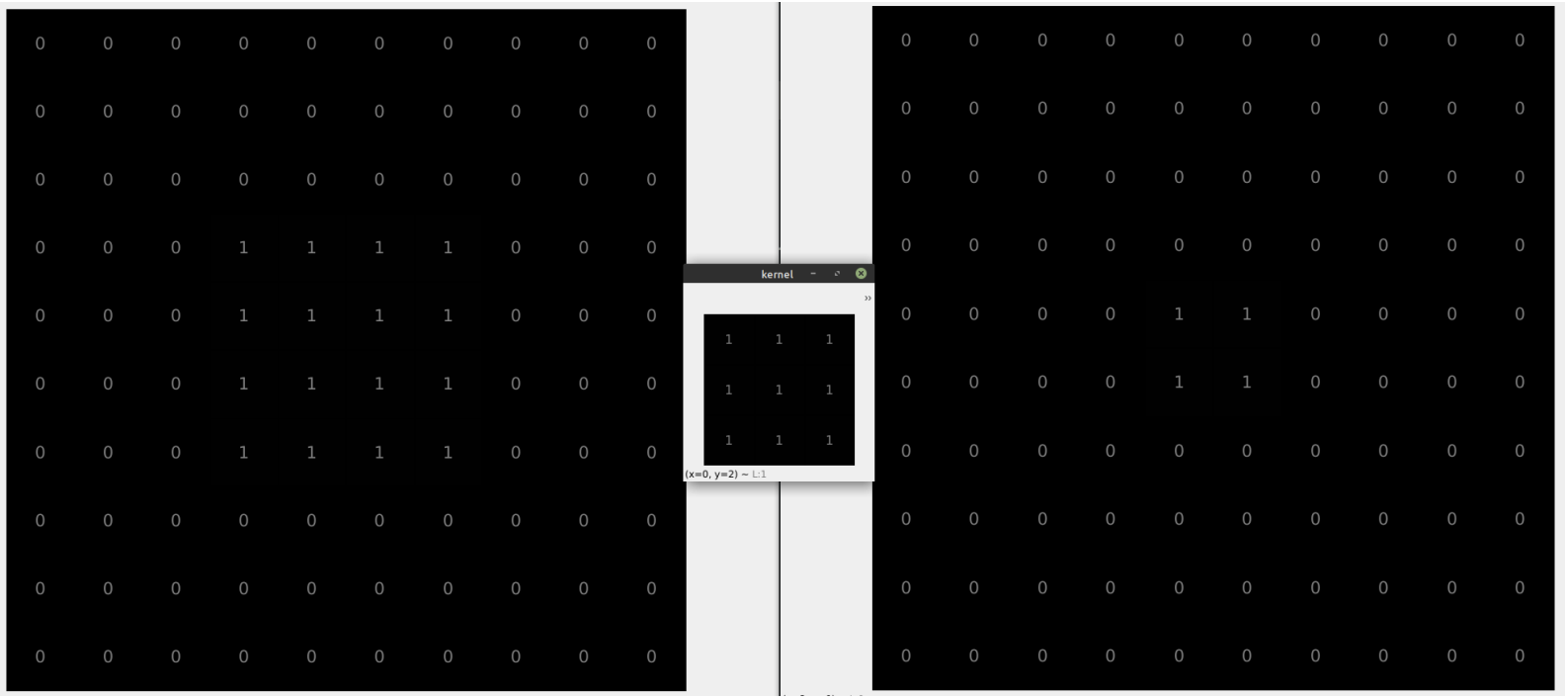
Morphological Operation

- The main principle is that the kernel is moved through to the input image
- The corresponding pixels inside the kernel vs. input image are examined
- Based on the operation (erosion, dilation), the pixels are for example eroded (made to zero)



Erosion

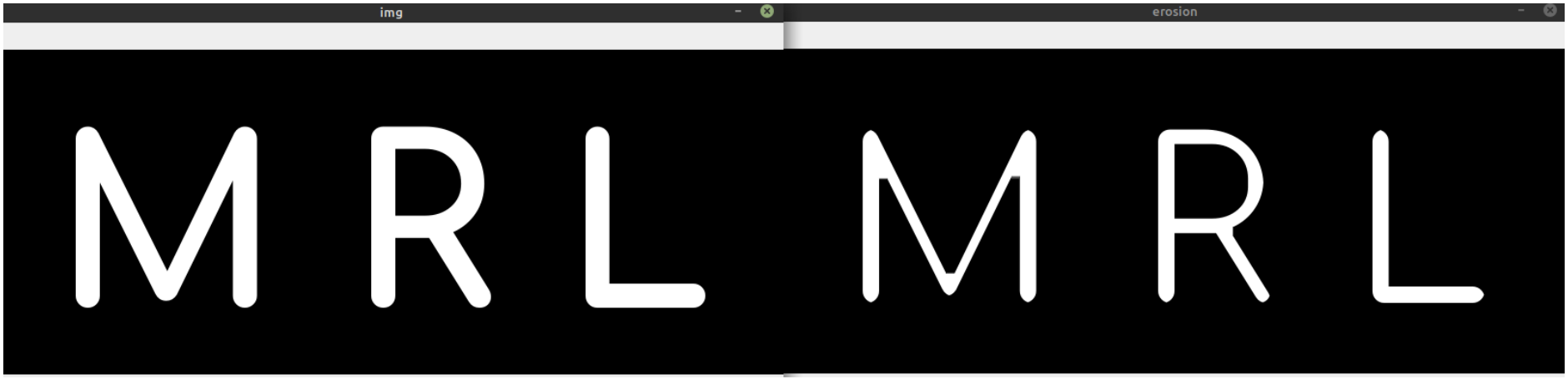
A pixel in the original image will be considered 1 only if all the pixels under the kernel is 1, otherwise it is eroded (made to zero). AND operator.



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Erosion

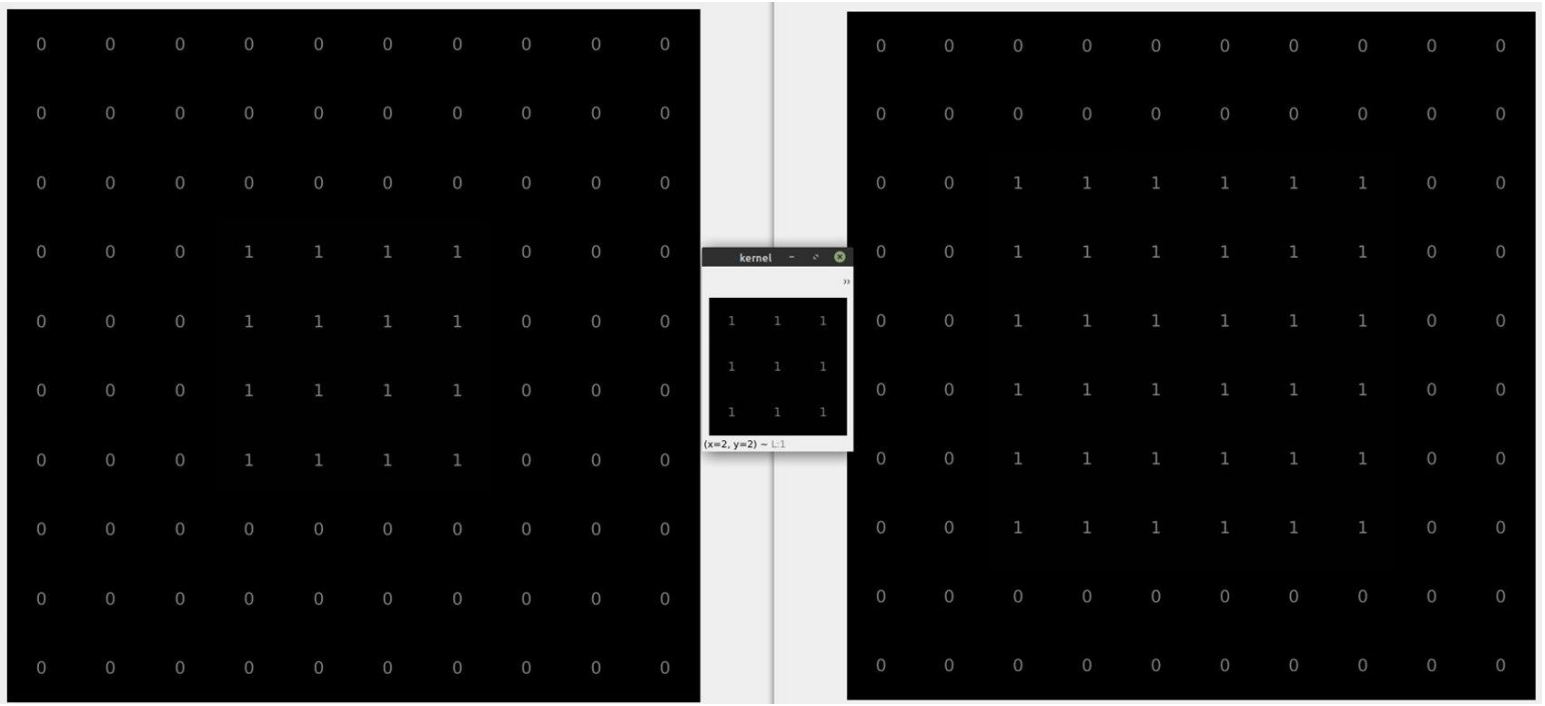
A pixel in the original image will be considered 1 only if all the pixels under the kernel is 1, otherwise it is eroded (made to zero).



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Dilation

Opposite of erosion: a pixel element is '1' if at least one pixel under the kernel is '1'.



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

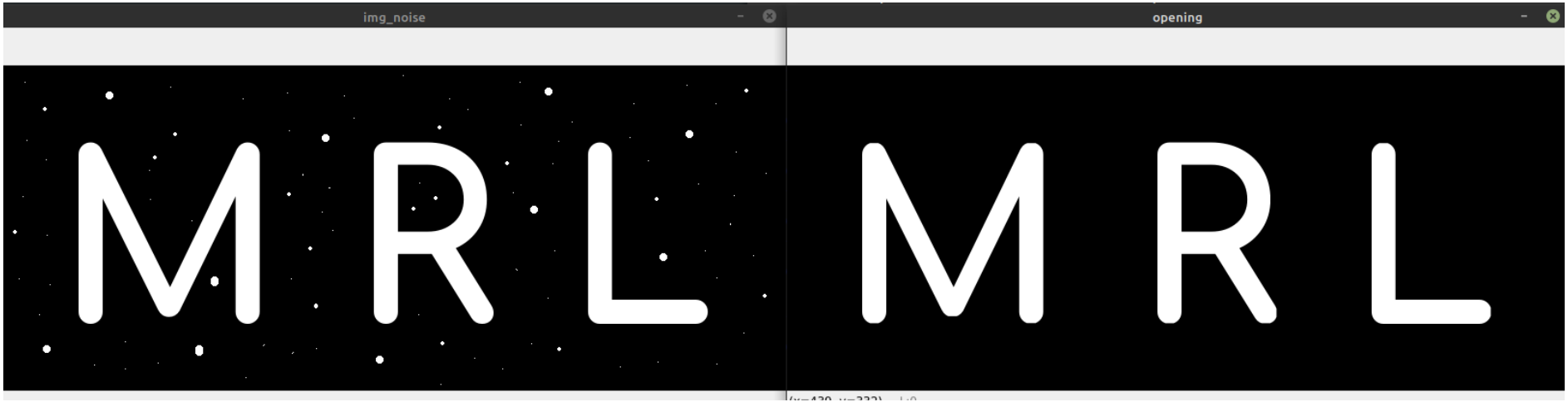
Dilation

Opposite of erosion: a pixel element is '1' if at least one pixel under the kernel is '1'.

https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Opening

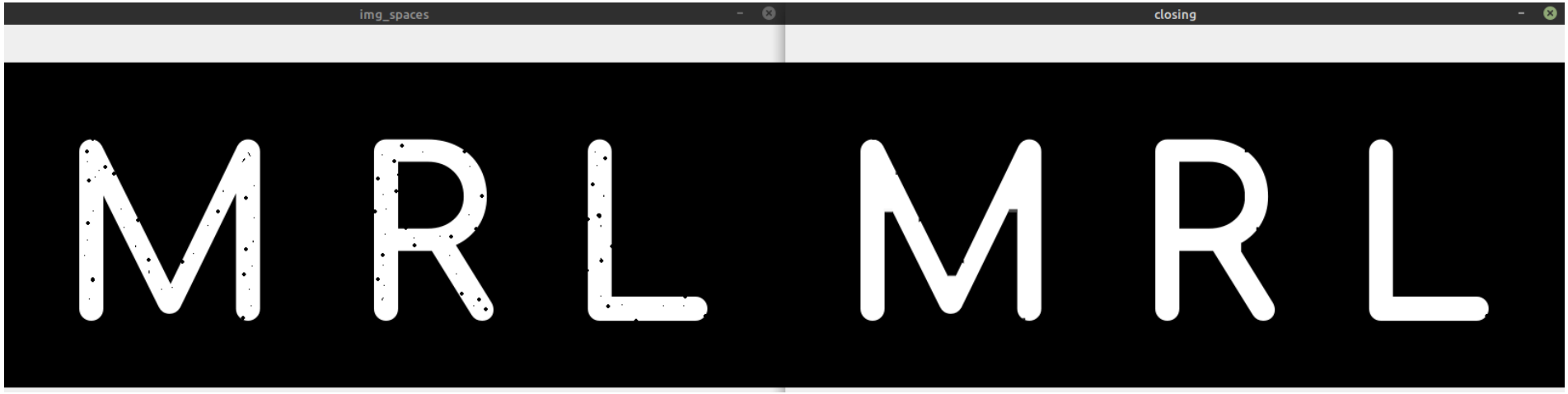
Opening: **erosion** followed by **dilation**



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Closing

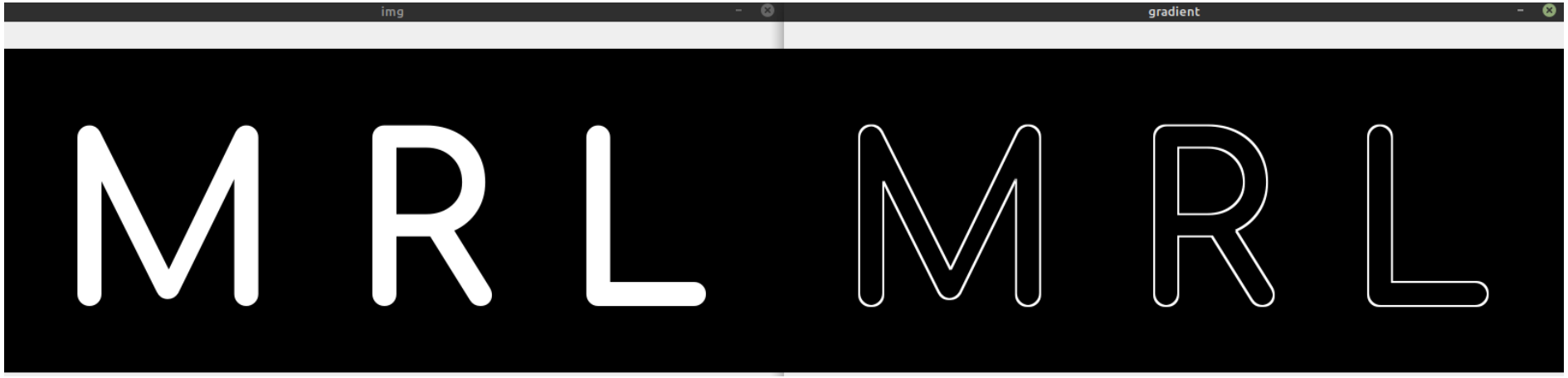
Opening: **dilation** followed by **erosion**



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Morphological Gradient

Difference between **dilation** and **erosion** of an image



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Structuring Elements

```
kernel = np.ones((5,5), np.uint8)
```

```
[[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]]
```

```
kernel = cv.getStructuringElement(cv.MORPH_CROSS, (5,5))
```

```
[[0 0 1 0 0]
 [0 0 1 0 0]
 [1 1 1 1 1]
 [0 0 1 0 0]
 [0 0 1 0 0]]
```

```
kernel = cv.getStructuringElement(cv.MORPH_RECT, (5,5))
```

```
[[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]]
```


https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Structuring Elements

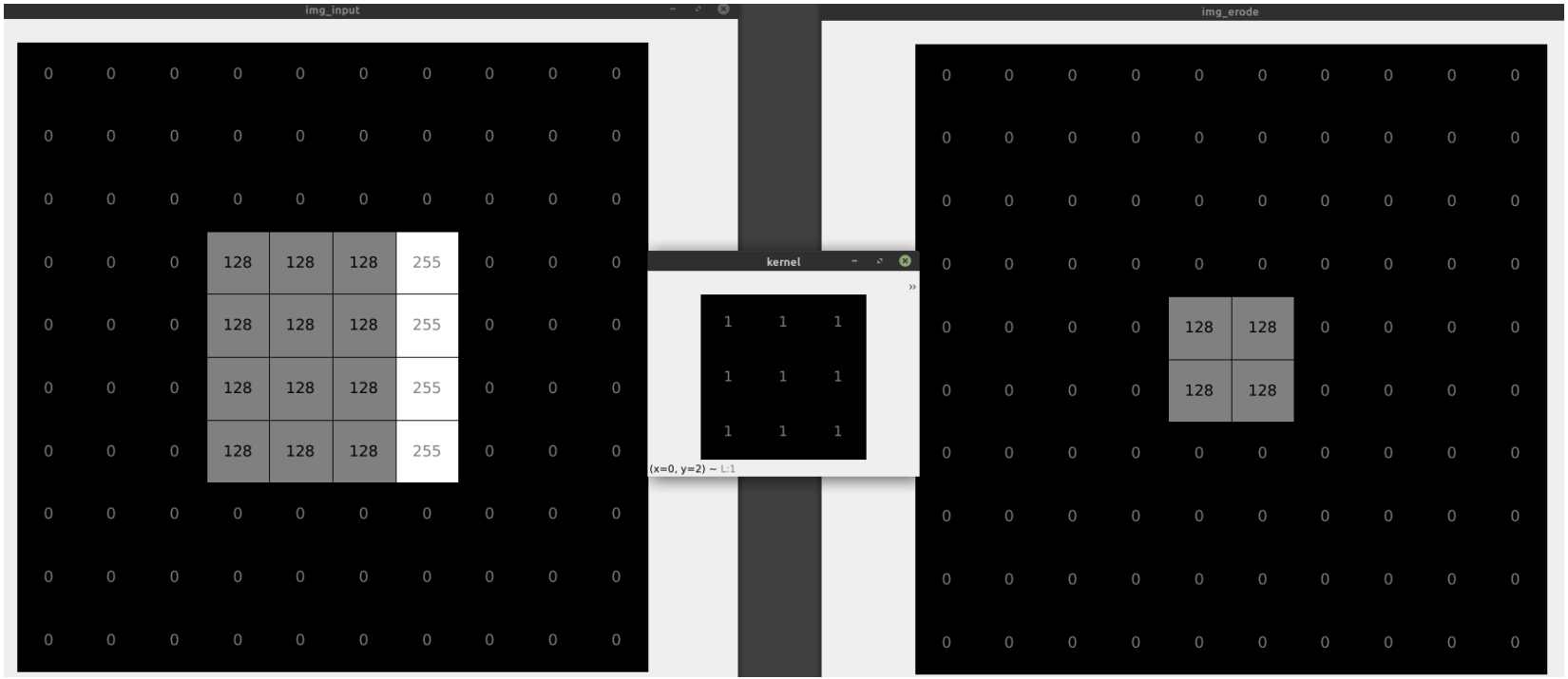
```
kernel = cv.getStructuringElement(cv.MORPH_ELLIPSE, (15,15))
```

```
[[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
 [0 0 0 1 1 1 1 1 1 1 1 0 0 0]
 [0 0 1 1 1 1 1 1 1 1 1 1 1 0 0]
 [0 1 1 1 1 1 1 1 1 1 1 1 1 1 0]
 [0 1 1 1 1 1 1 1 1 1 1 1 1 1 0]
 [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 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 1 1 1]
 [0 1 1 1 1 1 1 1 1 1 1 1 1 1 0]
 [0 1 1 1 1 1 1 1 1 1 1 1 1 1 0]
 [0 0 1 1 1 1 1 1 1 1 1 1 1 0 0]
 [0 0 0 1 1 1 1 1 1 1 1 1 0 0 0]
 [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]]
```


https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Grayscale/Color

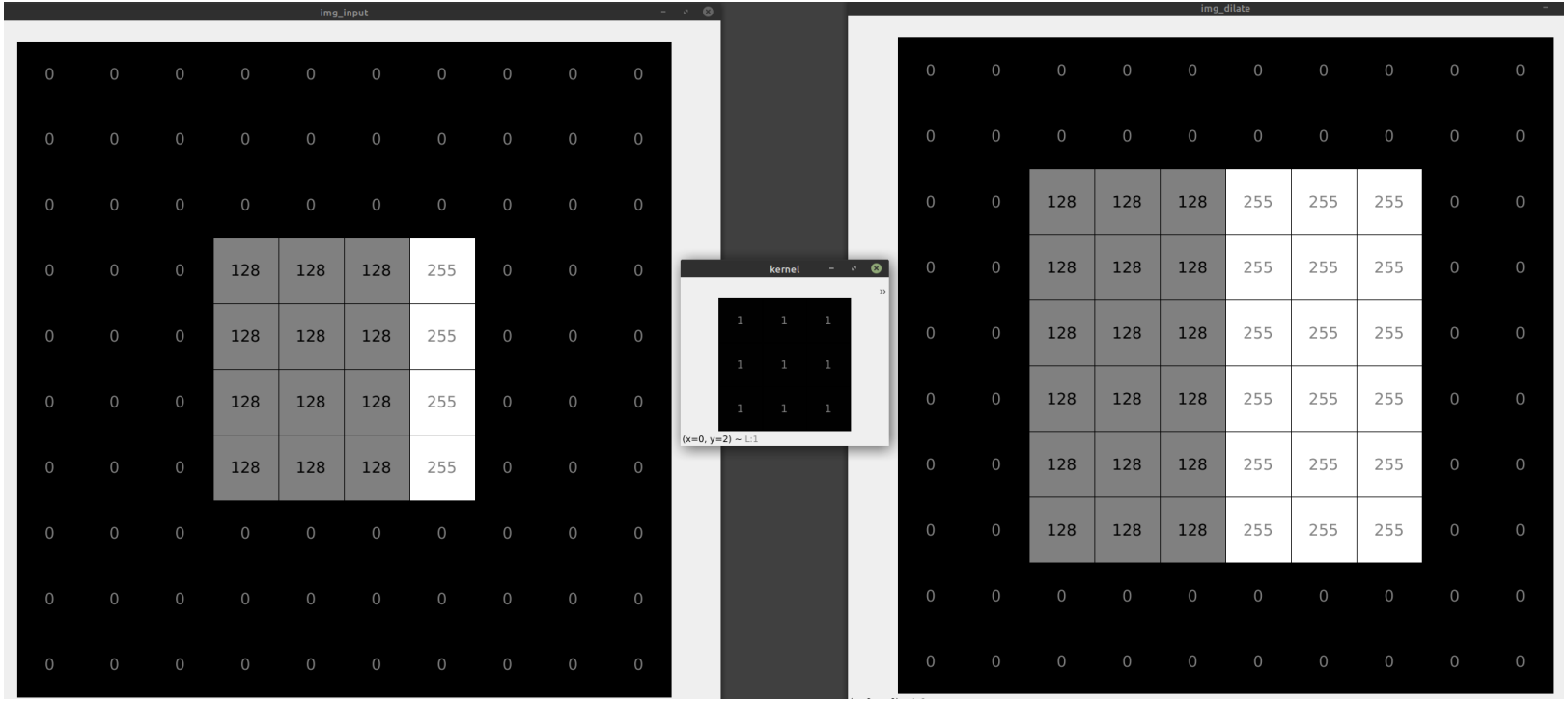
Idea of binary morphology can be extended to gray/color images with the use of max (Dilation) and min (Erosion) operation



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Grayscale/Color

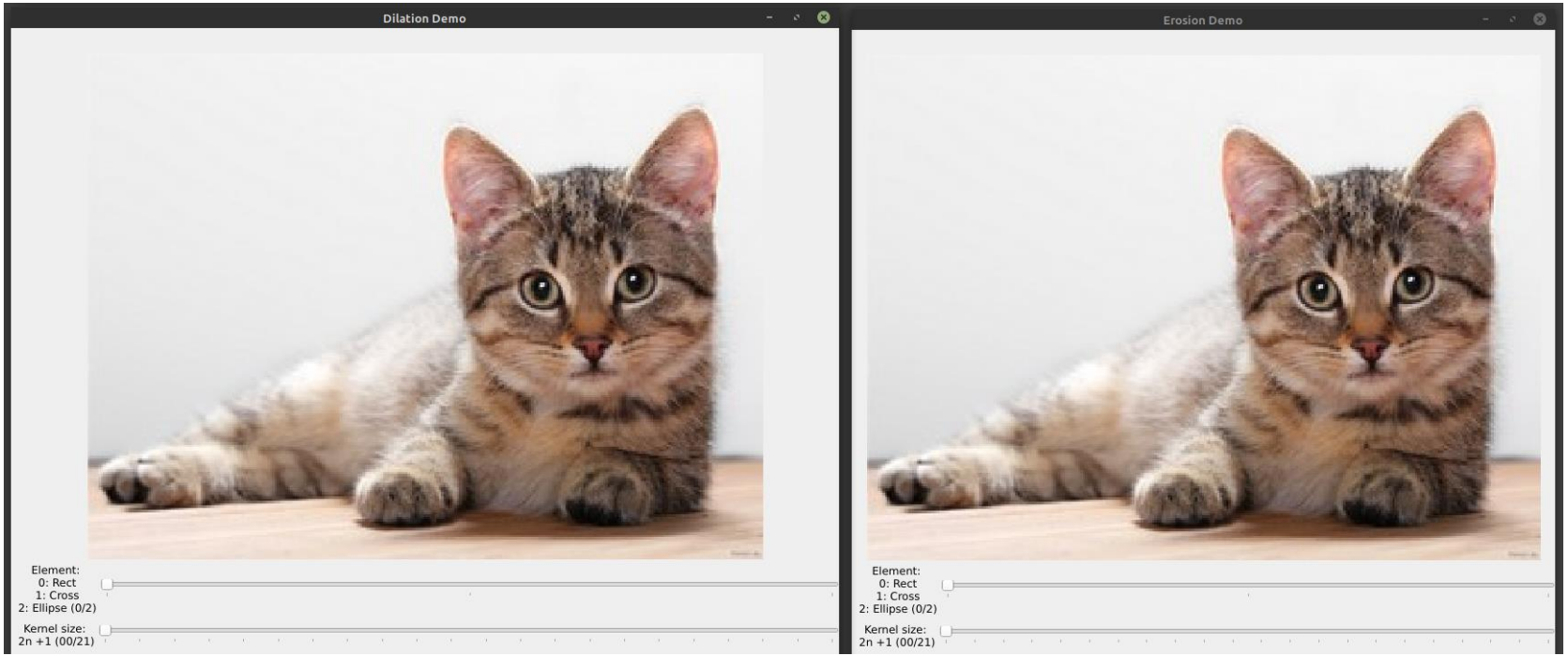
Idea of binary morphology can be extended to gray/color images with the use of max (Dilation) and min (Erosion) operation



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Grayscale/Color

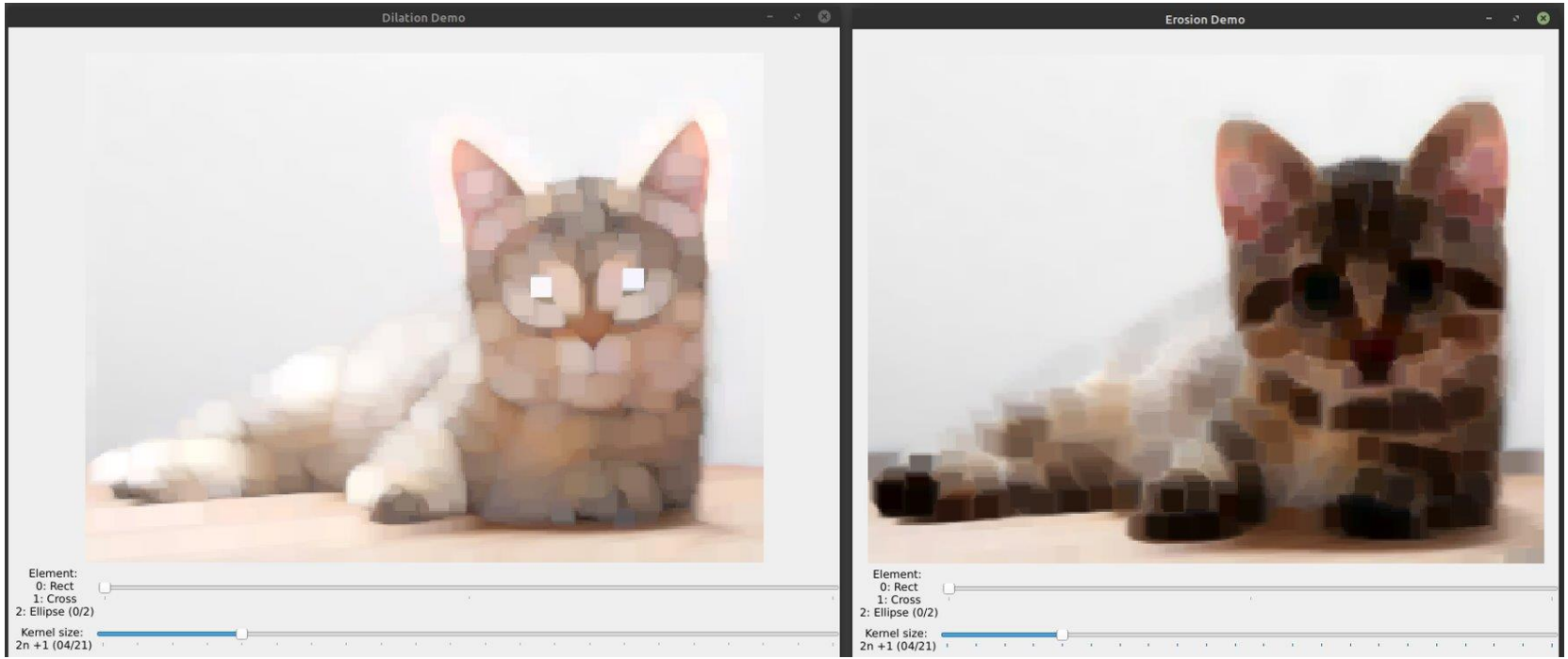
Idea of binary morphology can be extended to gray/color images with the use of max (dilation) and min (erosion) operation. Dilation and erosion change the brightness.
dilation - increased overall brightnes vs. erosion - dims the image



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Grayscale/Color

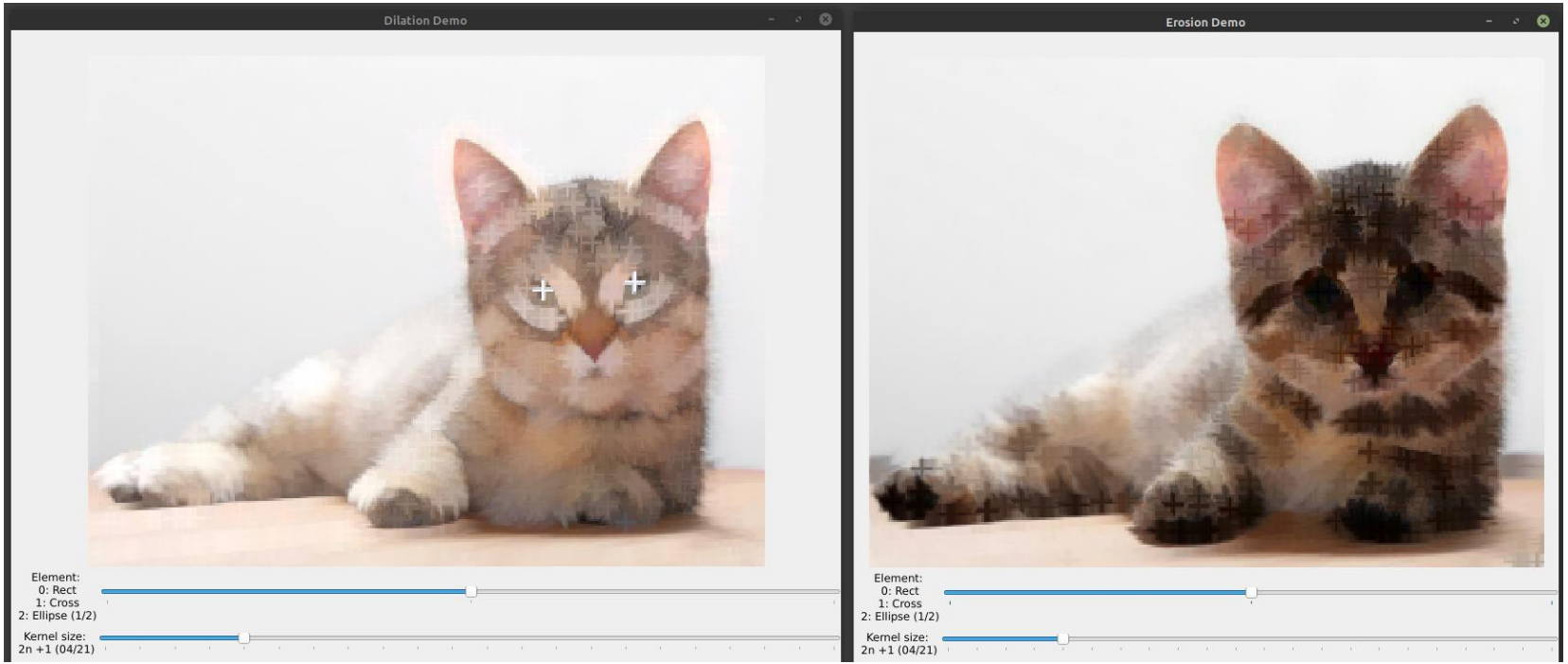
Idea of binary morphology can be extended to gray/color images with the use of max (dilation) and min (erosion) operation. Dilation and erosion change the brightness.
dilation - increased overall brightnes vs. erosion - dims the image



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Grayscale/Color

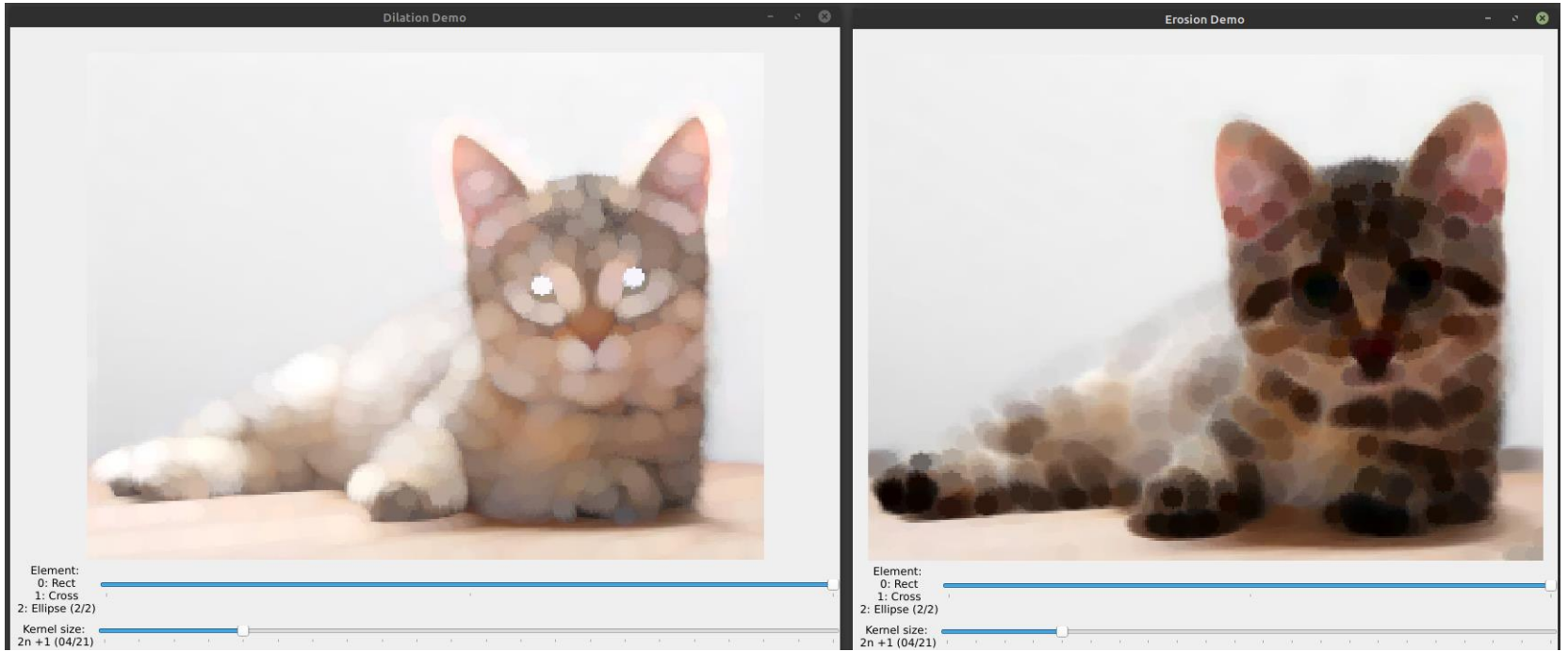
Idea of binary morphology can be extended to gray/color images with the use of max (dilation) and min (erosion) operation. Dilation and erosion change the brightness.
dilation - increased overall brightnes vs. erosion - dims the image



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Grayscale/Color

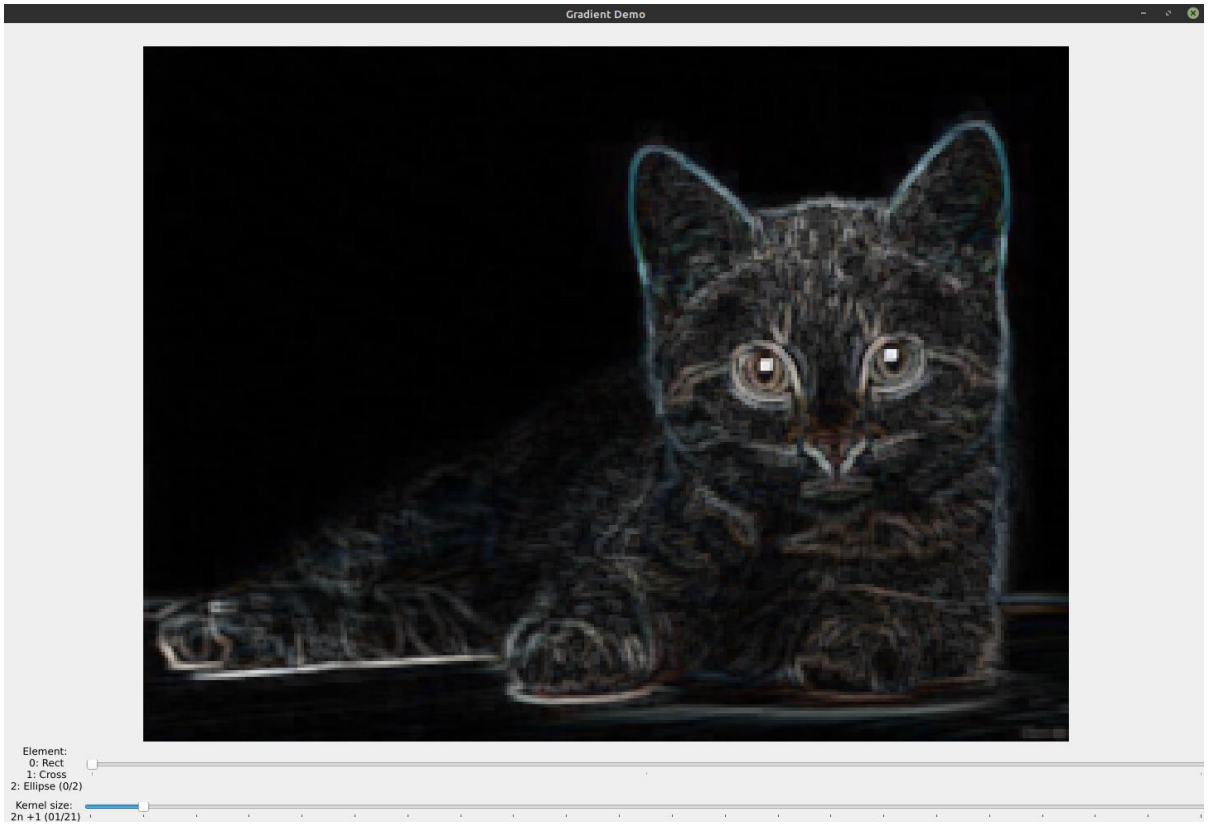
Idea of binary morphology can be extended to gray/color images with the use of max (dilation) and min (erosion) operation. Dilation and erosion change the brightness.
dilation - increased overall brightnes vs. erosion - dims the image



https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Grayscale/Color

Analogous way: a morphological gradient is the difference between a dilation and an erosion

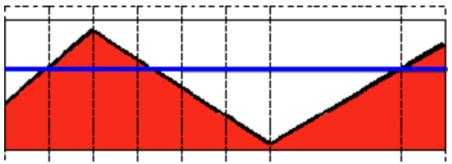


https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html https://docs.opencv.org/4.7.0/db/df6/tutorial_erosion_dilatation.html

Thresholding + Morph.

Types of Thresholding

- OpenCV offers the function **cv::threshold** to perform thresholding operations.
- We can effectuate 5 types of Thresholding operations with this function. We will explain them in the following subsections.
- To illustrate how these thresholding processes work, let's consider that we have a source image with pixels with intensity values $src(x, y)$. The plot below depicts this. The horizontal blue line represents the threshold $thresh$ (fixed).

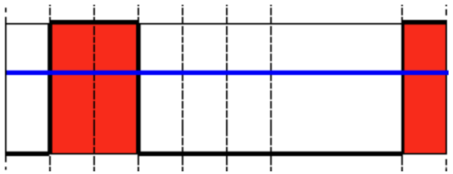


Threshold Binary

- This thresholding operation can be expressed as:

$$\texttt{dst}(x, y) = \begin{cases} \texttt{maxVal} & \text{if } \texttt{src}(x, y) > \texttt{thresh} \\ 0 & \text{otherwise} \end{cases}$$

- So, if the intensity of the pixel $src(x, y)$ is higher than $thresh$, then the new pixel intensity is set to a $MaxVal$. Otherwise, the pixels are set to 0.



https://docs.opencv.org/4.x/d7/d4d/tutorial_py_thresholding.html https://docs.opencv.org/4.7.0/db/d8e/tutorial_threshold.html https://docs.opencv.org/4.x/d3/db4/tutorial_py_watershed.html

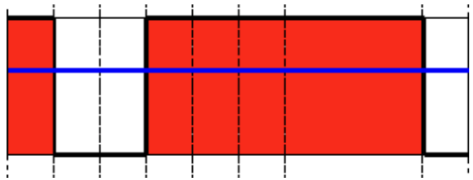
Thresholding + Morph.

Threshold Binary, Inverted

- This thresholding operation can be expressed as:

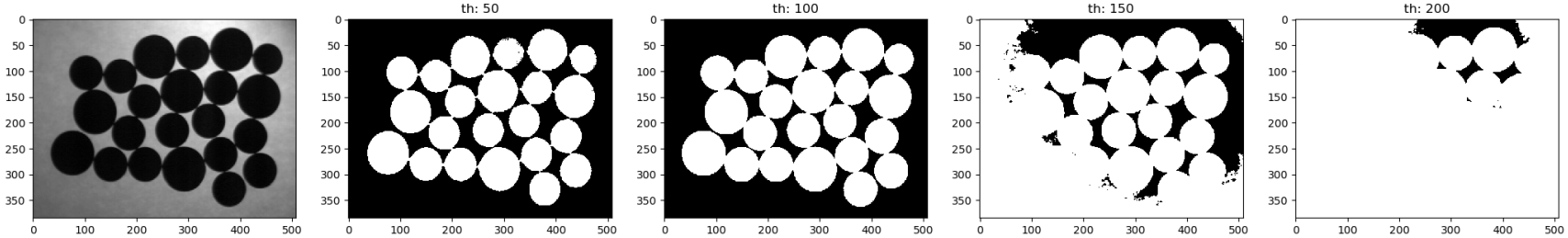
$$\texttt{dst}(x,y) = \begin{cases} 0 & \text{if } \texttt{src}(x,y) > \texttt{thresh} \\ \texttt{maxVal} & \text{otherwise} \end{cases}$$

- If the intensity of the pixel $src(x,y)$ is higher than $thresh$, then the new pixel intensity is set to a 0. Otherwise, it is set to $MaxVal$.



th_lst = [50, 100, 150, 200]

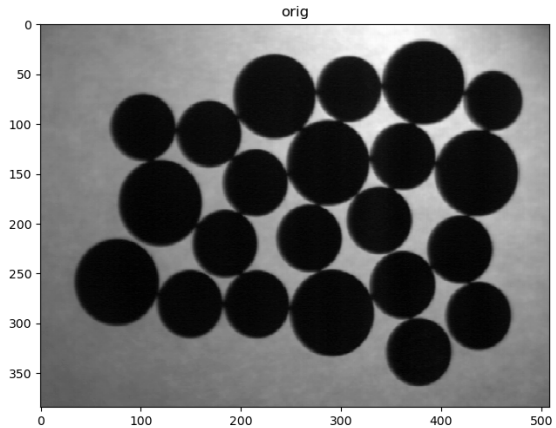
cv.threshold(img_input, th_lst[i], 255, cv.THRESH_BINARY_INV)

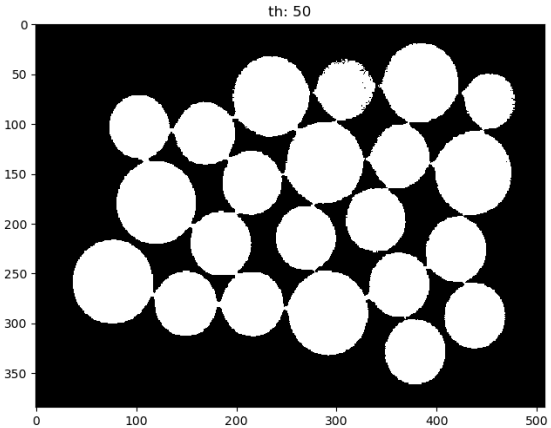


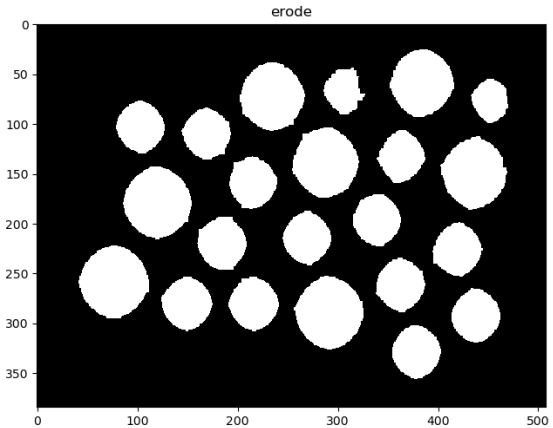
https://docs.opencv.org/4.x/d7/d4d/tutorial_py_thresholding.html https://docs.opencv.org/4.7.0/db/d8e/tutorial_threshold.html https://docs.opencv.org/4.x/d3/db4/tutorial_py_watershed.html

Thresholding + Morph.

```
img_input = cv.imread('mon1.png', 0)
kernel = cv.getStructuringElement(cv.MORPH_RECT, (11, 11))
ret, img_th = cv.threshold(img_input, 50, 255, cv.THRESH_BINARY_INV)
img_erode = cv.erode(img_th, kernel, iterations = 1)
```







https://docs.opencv.org/4.x/d7/d4d/tutorial_py_thresholding.html https://docs.opencv.org/4.7.0/db/d8e/tutorial_threshold.html https://docs.opencv.org/4.x/d3/db4/tutorial_py_watershed.html