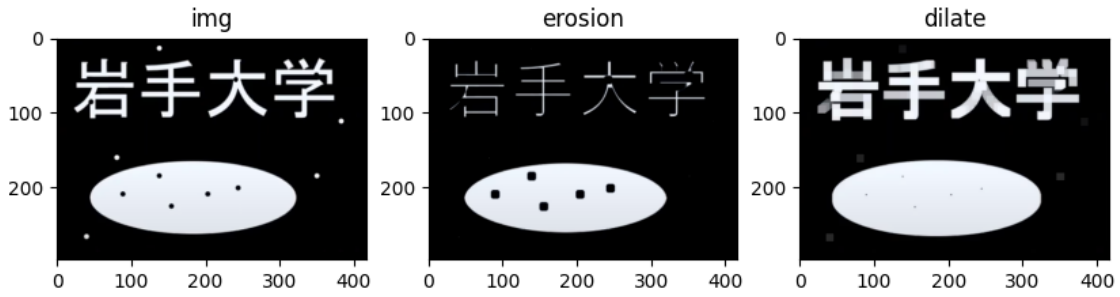
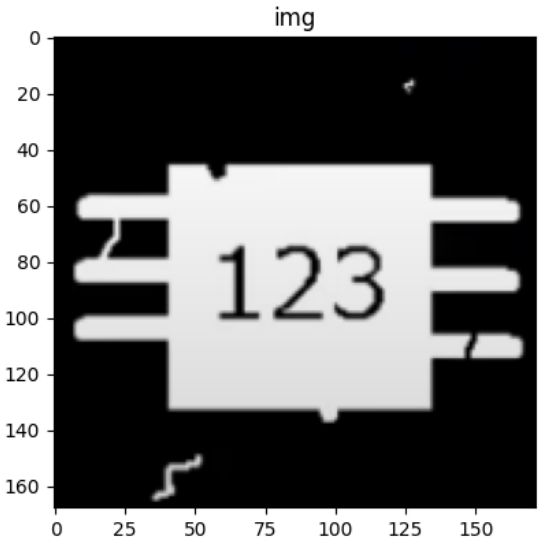
erode_dilate

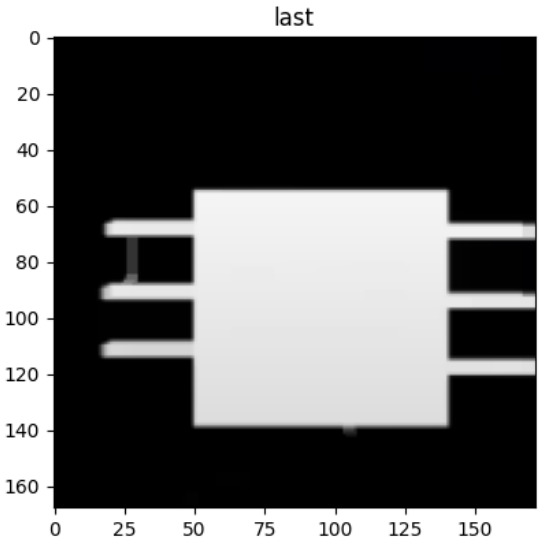
It is great to remove the noise of the picture using a simple method.

```
import cv2 as cv
import matplotlib.pyplot as plt
import numpy as np

img = cv.imread('images/01.png')

kernel1 = np.ones((6, 6), np.uint8)
kernel2 = np.ones((5, 5), np.uint8)

erosion = cv.erode(img, kernel1, iterations=1)
# Apply dilation to restore the background
dilate = cv.dilate(erosion, kernel1, iterations=1)
dilate = cv.dilate(dilate, kernel2, iterations=1)

fig, axes = plt.subplots(nrows=1, ncols=3, figsize=(10, 8),
dpi=100)
axes[0].imshow(img[:, :, ::-1])
axes[0].set_title("img")
axes[1].imshow(erosion[:, :, ::-1])
axes[1].set_title("erosion")
axes[2].imshow(dilate[:, :, ::-1])
axes[2].set_title("dilate")
plt.show()
```




```
import cv2 as cv
import matplotlib.pyplot as plt
import numpy as np

img = cv.imread('images/04.png')
kernel1 = np.ones((5, 5), np.uint8)
dilate = cv.dilate(img, kernel1, iterations=1)
dilate = cv.dilate(dilate, kernel1, iterations=1)
erosion = cv.erode(dilate, kernel1, iterations=1)
erosion = cv.erode(erosion, kernel1, iterations=1)
erosion = cv.erode(erosion, kernel1, iterations=1)
dilate = cv.dilate(erosion, kernel1, iterations=1)

fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(10, 8),
dpi=100)
axes[0].imshow(img[:, :, ::-1])
axes[0].set_title("img")
axes[1].imshow(dilate[:, :, ::-1])
axes[1].set_title("last")
plt.show()
```


img
123


last