Dilation and Erosion

Dilation and erosion are known as **morphological operations**. They are often performed on binary images, similar to contour detection. Dilation enlarges bright, white areas in an image by adding pixels to the perceived boundaries of objects in that image. Erosion does the opposite: it removes pixels along object boundaries and shrinks the size of objects.

Often these two operations are performed in sequence to enhance important object traits!

Dilation

To dilate an image in OpenCV, you can use the `dilate` function and three inputs: an original binary image, a kernel that determines the size of the dilation (None will result in a default size), and a number of iterations to perform the dilation (typically = 1). In the below example, we have a 5x5 kernel of ones, which move over an image, like a filter, and turn a pixel white if any of its surrounding pixels are white in a 5x5 window! We'll use a simple image of the cursive letter "j" as an example.

```
# Reads in a binary image
image = cv2.imread('j.png', 0)

# Create a 5x5 kernel of ones
kernel = np.ones((5,5),np.uint8)

# Dilate the image
```

```
dilation = cv2.dilate(image, kernel, iterations = 1)
```

Erosion

To erode an image, we do the same but with the `erode` function.

```
# Erode the image
erosion = cv2.erode(image, kernel, iterations = 1)
```

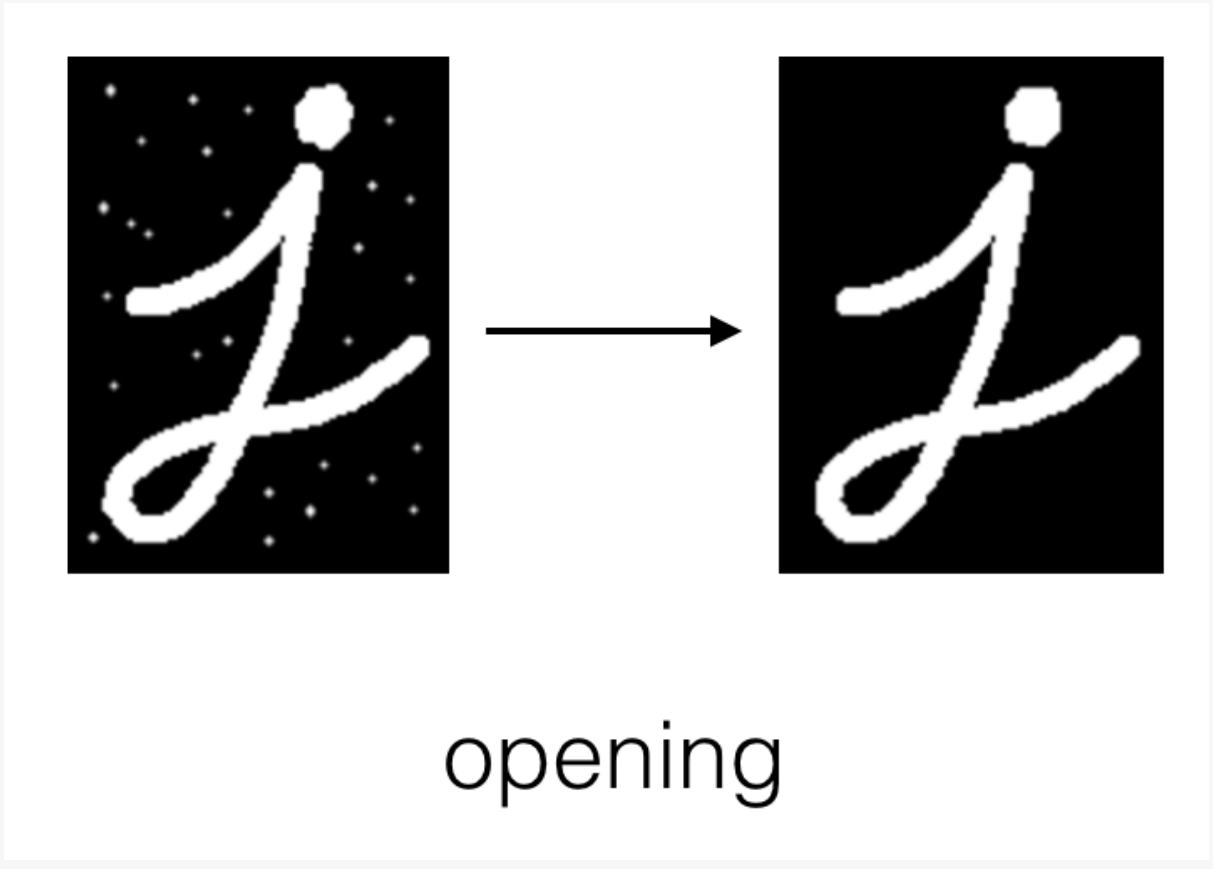
The letter "j": (left) erosion, (middle) original image, (right) dilation

Opening

As mentioned, above, these operations are often *combined* for desired results! One such combination is called **opening**, which is **erosion followed by dilation**. This is useful in noise reduction in which erosion first gets rid of noise (and shrinks the object) then dilation enlarges the object again, but the noise will have disappeared from the previous erosion!

To implement this in OpenCV, we use the function `morphologyEx` with our original image, the operation we want to perform, and our kernel passed in.

```
opening = cv2.morphologyEx(image, cv2.MORPH_OPEN, kernel)
```



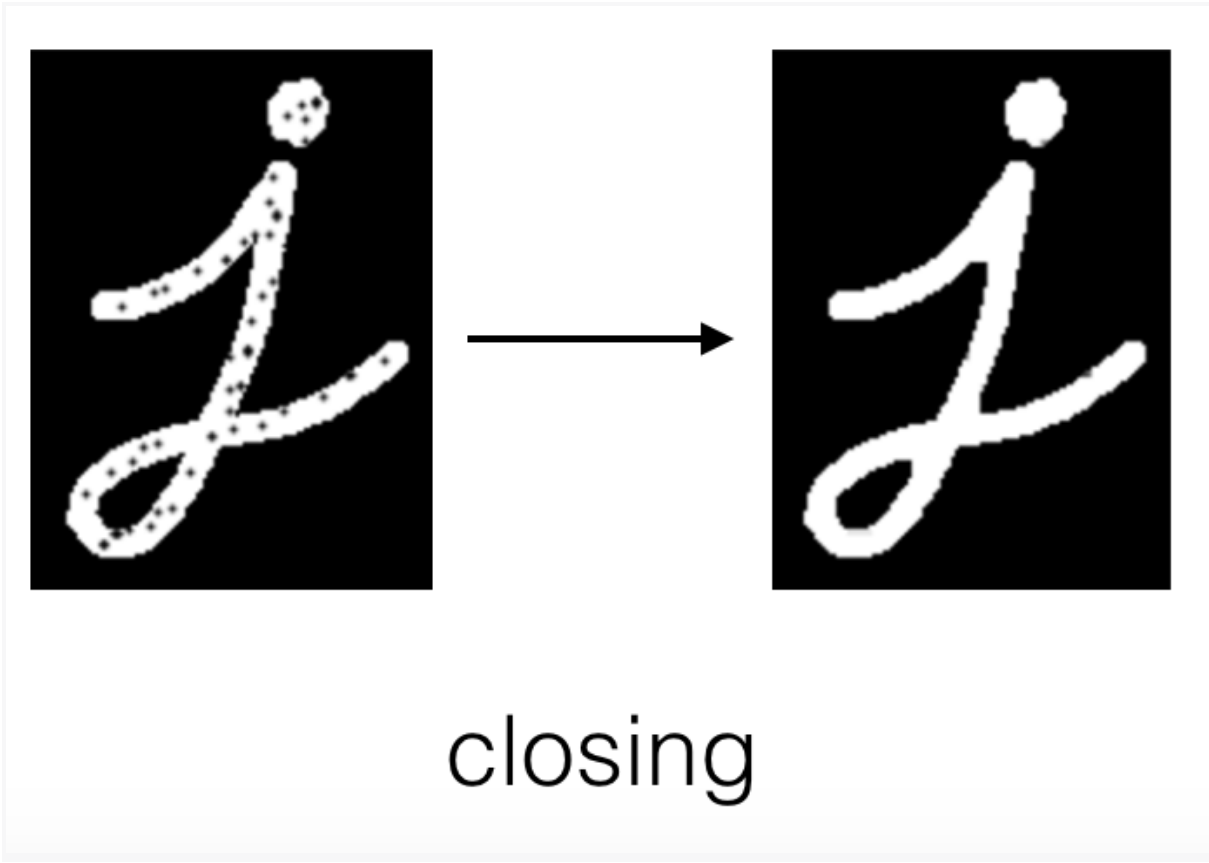
Opening

Closing

Closing is the reverse combination of opening; it's **dilation followed by erosion**, which is useful in *closing* small holes or dark areas within an object.

Closing is reverse of Opening, Dilation followed by Erosion. It is useful in closing small holes inside the foreground objects, or small black points on the object.

```
closing = cv2.morphologyEx(img, cv2.MORPH_CLOSE, kernel)
```



Closing

Many of these operations try to extract better (less noisy) information about the shape of an object or enlarge important features, as in the case of corner detection!