

# image-processing

January 20, 2024

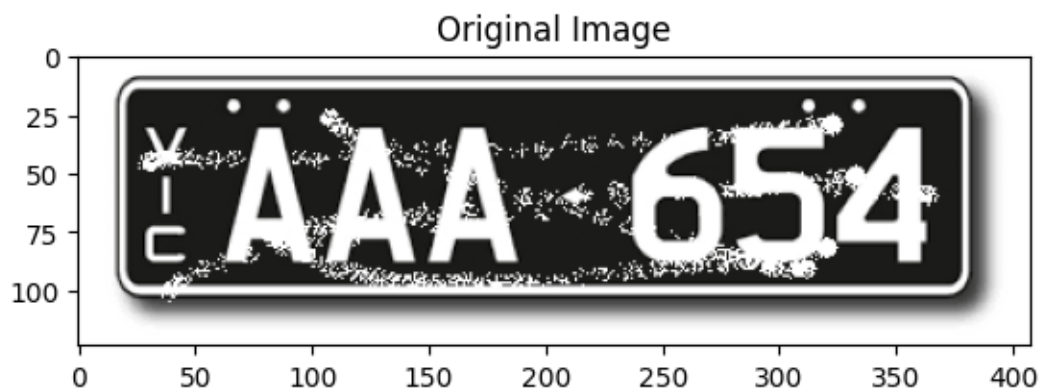
```
[1]: import numpy as np
import cv2
import matplotlib as mpl
import matplotlib.pyplot as plt
```

```
[3]: img=plt.imread("/kaggle/input/images/image.png")
```

```
[4]: print(img.shape) # printing shape of image. It is colour so 124 x 408 pixel
↳having 3 channel - RGB
```

(124, 408, 4)

```
[5]: plt.imshow(img)
plt.title("Original Image")
plt.show()
```



## 0.1 CV2 - Image Processing

Removing noise from images

Morphological analysis

Image rotation and flipping

# 1 Morphological Analysis

Erosion: loss of pixels from the edges

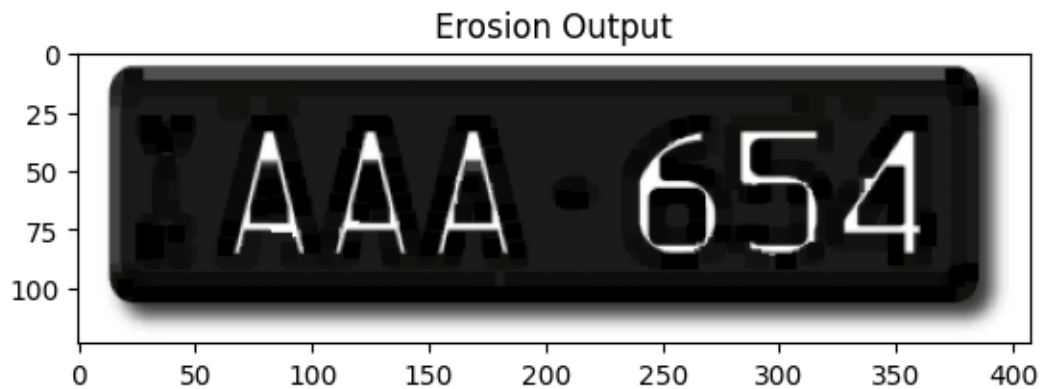
Dilation: gain of pixels on the edges

## 1.0.1 Erosion

```
[6]: kernel = np.ones((7,7))
img2=cv2.erode(img,kernel)

print(img.shape)
plt.imshow(img2)
plt.title("Erosion Output")
plt.show()
```

(124, 408, 4)



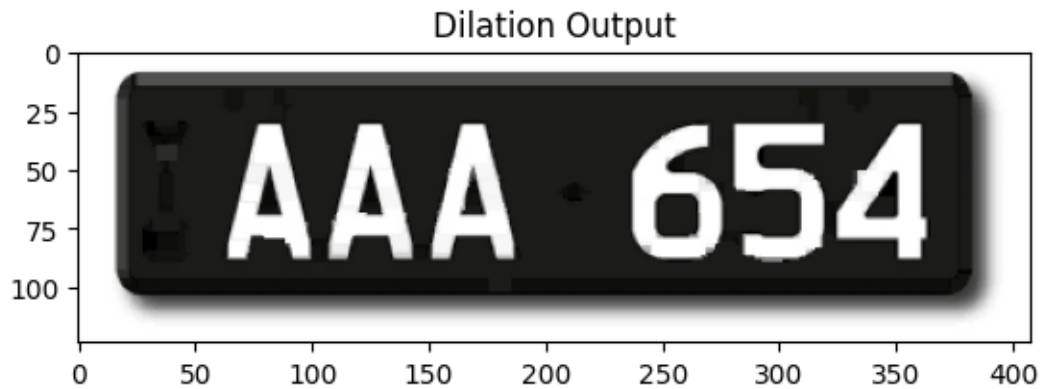
## 1.0.2 Dilation

```
[7]: kernel = np.ones((7,7))

img3=cv2.dilate(img2,kernel) # we are using the image on which we already
    ↪ applied Erosion

print(img.shape)
plt.imshow(img3)
plt.title("Dilation Output")
plt.show()
```

(124, 408, 4)



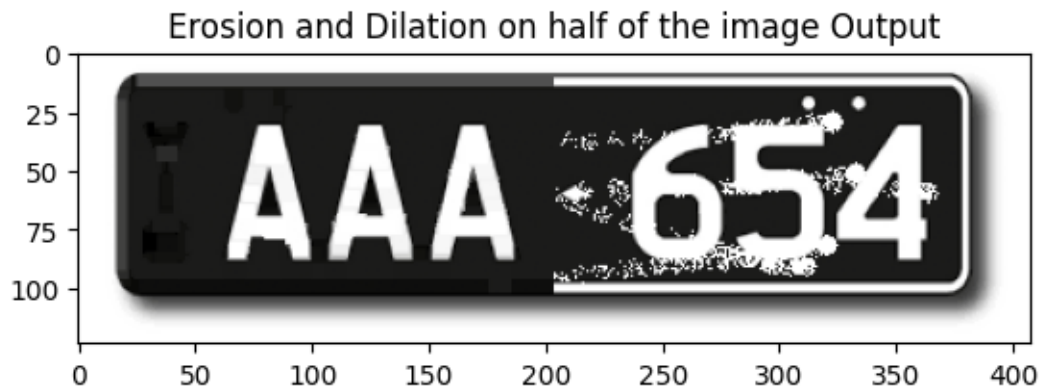
### 1.0.3 Erosion and Dilation Both

```
[8]: # Apply Erosion and Dilation on half of the image

img4=img.copy()
img4[:, :204]=cv2.dilate(cv2.erode(img4[:, :204],kernel),kernel)

print(img.shape)
plt.imshow(img4)
plt.title("Erosion and Dilation on half of the image Output")
plt.show()
```

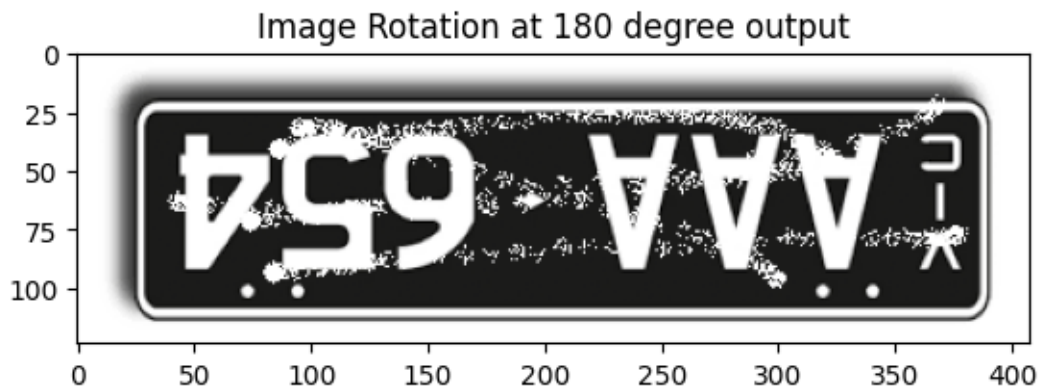
(124, 408, 4)



#### 1.0.4 image Rotation

```
[10]: img6=cv2.rotate(img,cv2.ROTATE_180) # Other options: ROTATE_90_CLOCKWISE and
      ↪ROTATE_90_ANTICLOCKWISE
print(img6.shape)
plt.imshow(img6)
plt.title("Image Rotation at 180 degree output")
plt.show()
```

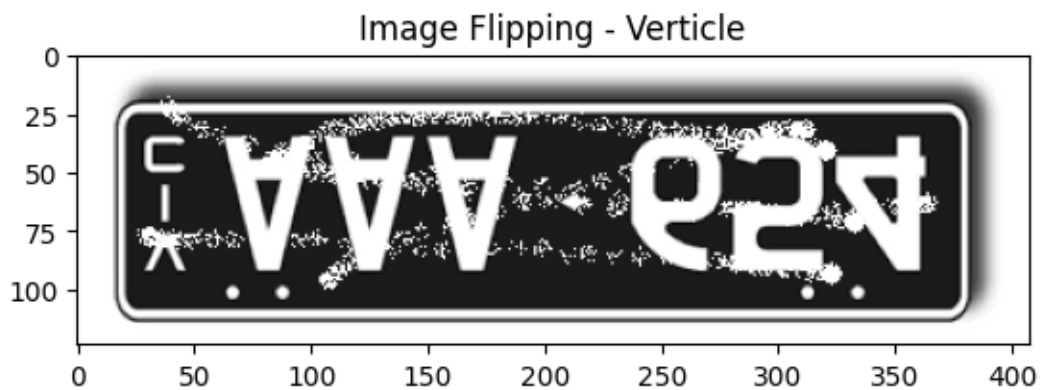
(124, 408, 4)



#### 1.0.5 Image flipping - Verticle

```
[11]: img7=cv2.flip(img,0) # 0 is for verticle flipping
print(img7.shape)
plt.imshow(img7)
plt.title("Image Flipping - Verticle")
plt.show()
```

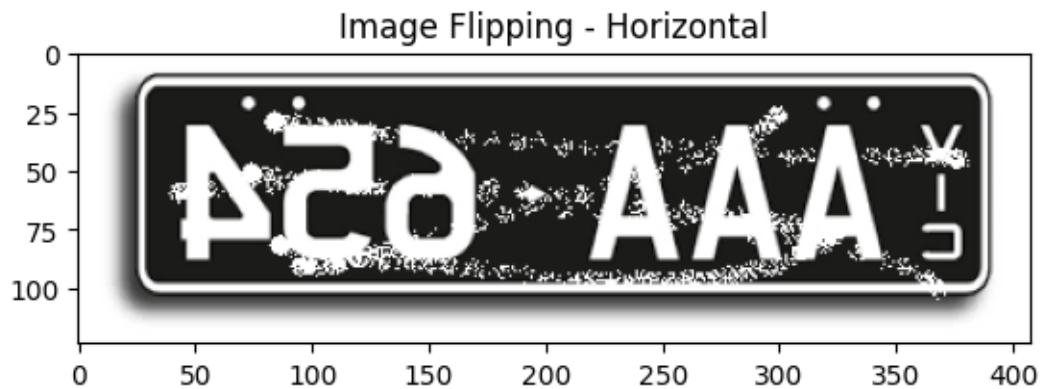
(124, 408, 4)



### 1.0.6 Image flipping - Horizontal

```
[12]: img8=cv2.flip(img,1) # 1 is for horizontal flipping
      print(img8.shape)
      plt.imshow(img8)
      plt.title("Image Flipping - Horizontal")
      plt.show()
```

(124, 408, 4)



### 1.0.7 Image flipping - Both (horizontal and vertical)

```
[13]: img9=cv2.flip(img,-1) # -1 is for vertical and horizontal both flipping
      print(img9.shape)
      plt.imshow(img9)
      plt.title("Image Flipping - Vertical and Horizontal")
      plt.show()
```

(124, 408, 4)

