Smooting Images

Averaging

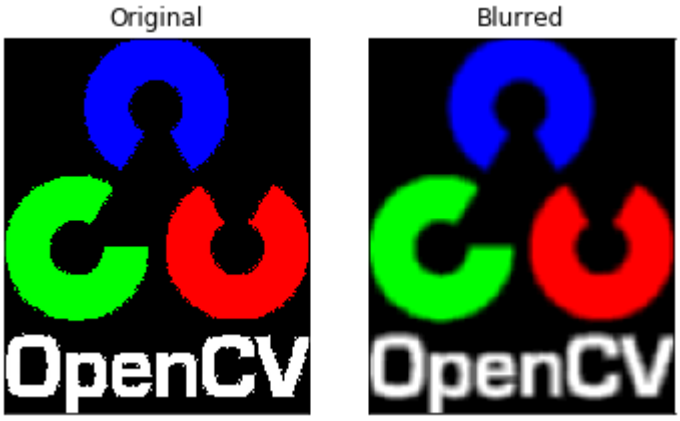
In [16]:

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('images/opencv.png')


blur = cv2.blur(img,(5,5))


plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur),plt.title('Blurred')
plt.xticks([]), plt.yticks([])
plt.show()
```



Gaussian Bluring

In [15]:

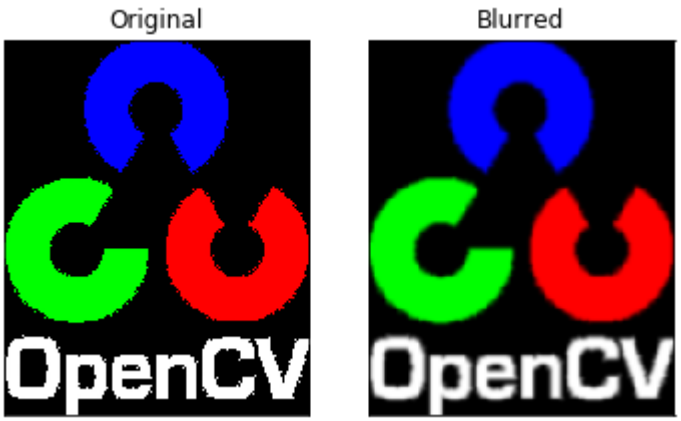
```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('images/opencv.png')


blur = cv2.GaussianBlur(img,(5,5),0)


plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur),plt.title('Blurred')
```

```
plt.xticks([]), plt.yticks([])
plt.show()
```



Median Filtering

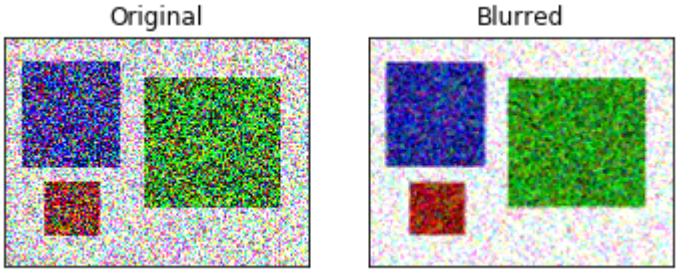
In [18]:

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('images/noise1.png')


median = cv2.medianBlur(img,5)


plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(median),plt.title('Blurred')
plt.xticks([]), plt.yticks([])
plt.show()
```



Morphological Transform

Erosion

In [23]:

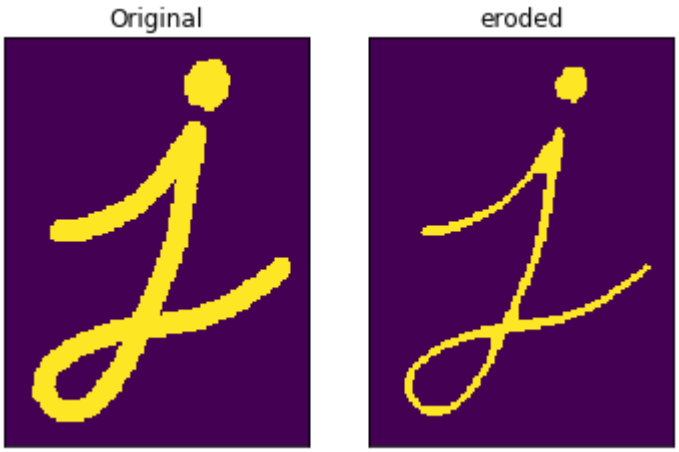
```
import cv2
import numpy as np

img = cv2.imread('images/j.png',0)


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

```
erosion = cv2.erode(img,kernel,iterations = 1)


plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(erosion),plt.title('eroded')
plt.xticks([]), plt.yticks([])
plt.show()
```



In [ ]:

Dilation

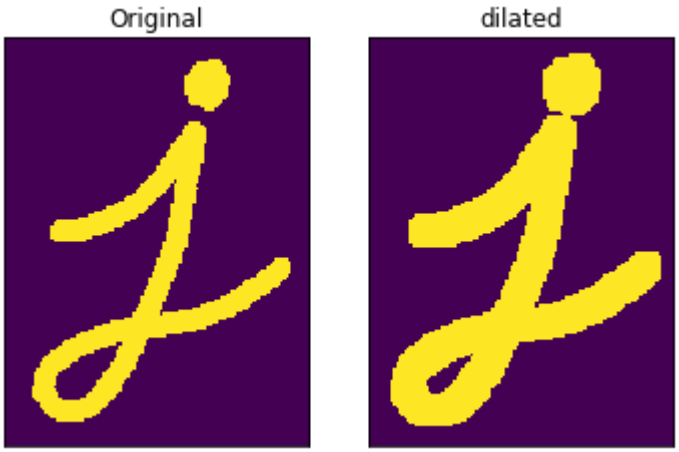
In [25]:

```
import cv2
import numpy as np

img = cv2.imread('images/j.png',0)


kernel = np.ones((5,5),np.uint8)
dilation = cv2.dilate(img,kernel,iterations = 1)


plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(dilation),plt.title('dilated')
plt.xticks([]), plt.yticks([])
plt.show()
```



In [ ]:

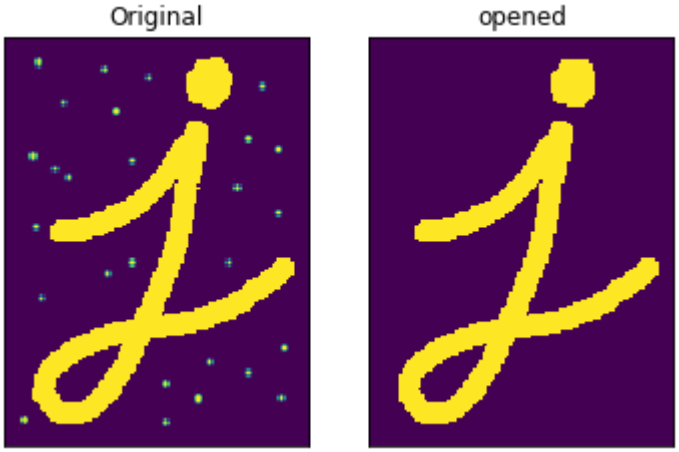
Opening

In [28]:
```
import cv2
import numpy as np

img = cv2.imread('images/j_dot.png',0)


kernel = np.ones((5,5),np.uint8)
opening = cv2.morphologyEx(img, cv2.MORPH_OPEN, kernel)


plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(opening),plt.title('opened')
plt.xticks([]), plt.yticks([])
plt.show()
```



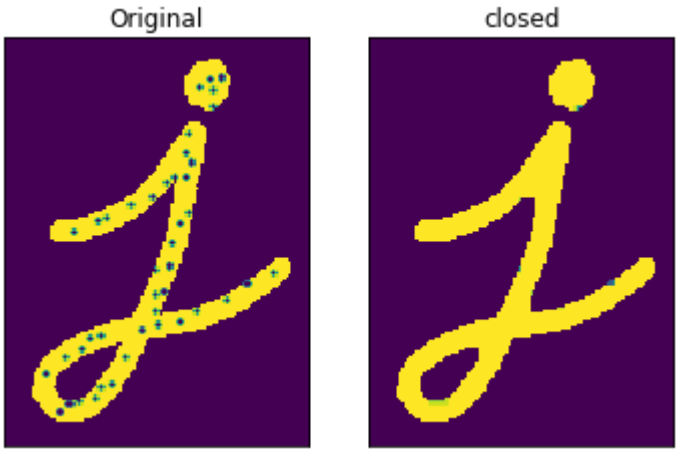
Closing

In [30]:
```
import cv2
import numpy as np
```

```
img = cv2.imread('images/j_dot_in.png',0)


kernel = np.ones((5,5),np.uint8)
closing = cv2.morphologyEx(img, cv2.MORPH_CLOSE, kernel)


plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(closing),plt.title('closed')
plt.xticks([]), plt.yticks([])
plt.show()
```



Morphological Gradient

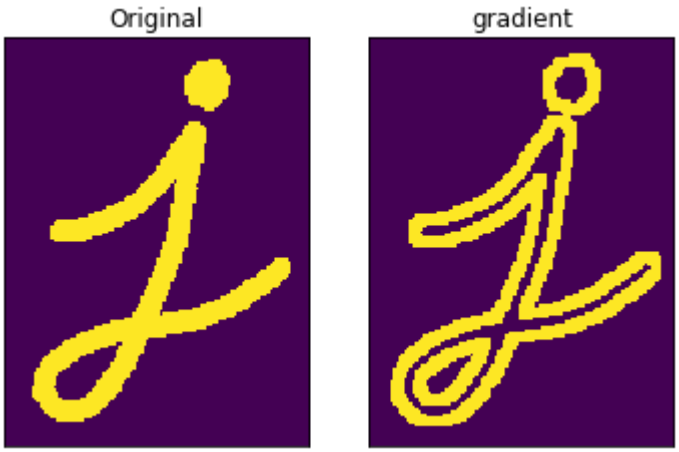
In [32]:

```
import cv2
import numpy as np

img = cv2.imread('images/j.png',0)


kernel = np.ones((5,5),np.uint8)
gradient = cv2.morphologyEx(img, cv2.MORPH_GRADIENT, kernel)


plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(gradient),plt.title('gradient')
plt.xticks([]), plt.yticks([])
plt.show()
```



Home Work

1) Using the provided HSV low and high values try to isolate the blue ball. Make sure the entire blue ball is visible

Solution - problem 1

In [2]:

```
import cv2
import numpy as np

img = cv2.imread('images/smarties.png',1)

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)


lower_blue = np.array([98,118,85],dtype = 'uint8')
upper_blue = np.array([136,241,255],dtype = 'uint8')


mask1 = cv2.inRange(hsv, lower_blue, upper_blue)
out1 = cv2.bitwise_and(img,img, mask = mask1)


kernel = np.ones((3,3),np.uint8)
mask11 = cv2.dilate(mask1,kernel,iterations = 4)
```

```
mask2 = cv2.erode(mask11,kernel,iterations = 4)
# mask2 = cv2.morphologyEx(img, cv2.MORPH_CLOSE, kernel)
out2 = cv2.bitwise_and(img,img, mask = mask2)


# cv2.imshow("img",img)
cv2.imshow("mask1",mask1)
cv2.imshow("out1",out1)
cv2.imshow("mask2",mask2)
cv2.imshow("out2",out2)


cv2.waitKey(0)
cv2.destroyAllWindows()
```