



BCA 607 Hareket Analizi Sistemleri

OpenCV ile Görüntü İşleme 2



SERDAR ARITAN

serdar.aritan@hacettepe.edu.tr

Bilgisayar Grafiği Anabilim Dalı

www.bil-grafik.hacettepe.edu.tr

Bilişim Enstitüsü

www.bilisim.hacettepe.edu.tr

Hacettepe Üniversitesi, Ankara, Türkiye

www.hacettepe.edu.tr



Yansıtıcı işaret yakalama

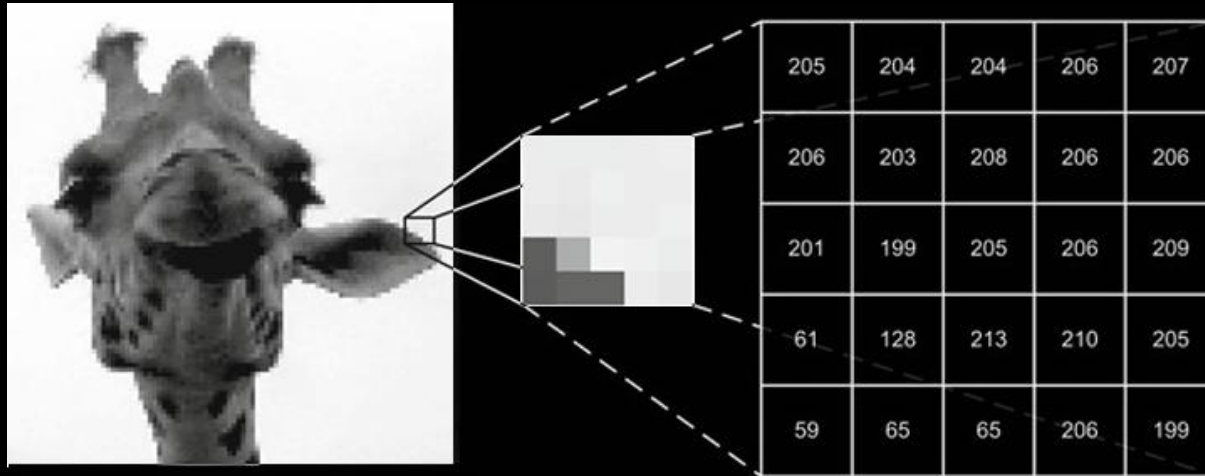




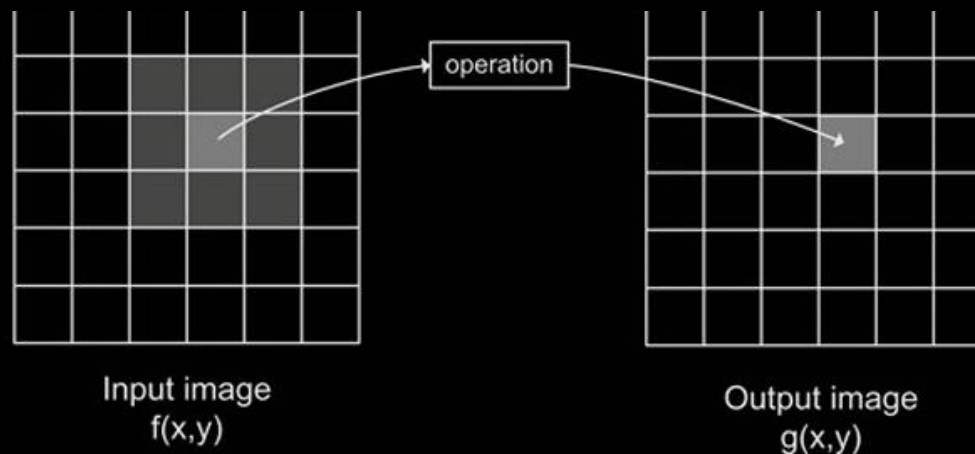
Yansıtıcı işaret yakalama



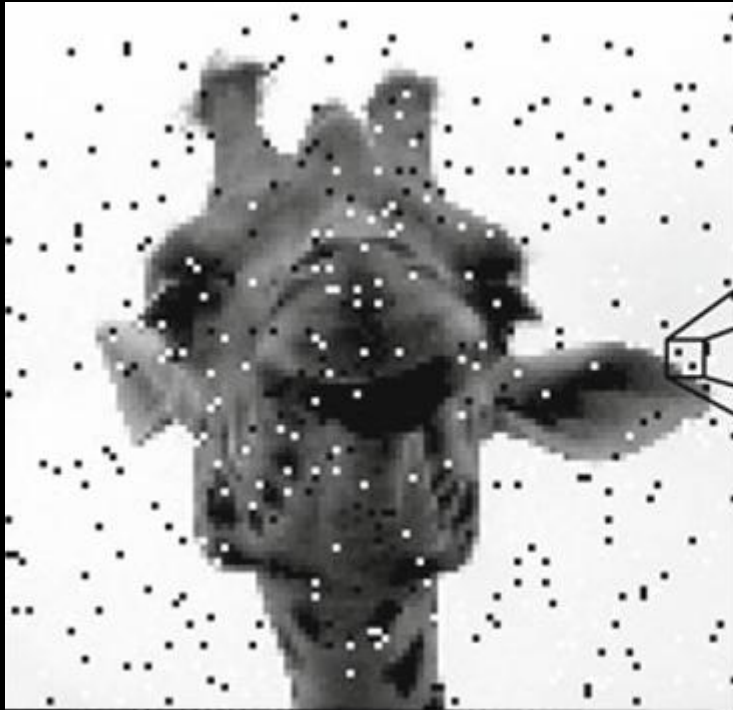
The Mean Filter



$$\text{Mean value} = \frac{205 + 204 + 204 + 206 + 0 + 208 + 201 + 199 + 205}{9} = 181.3$$



The Median Filter

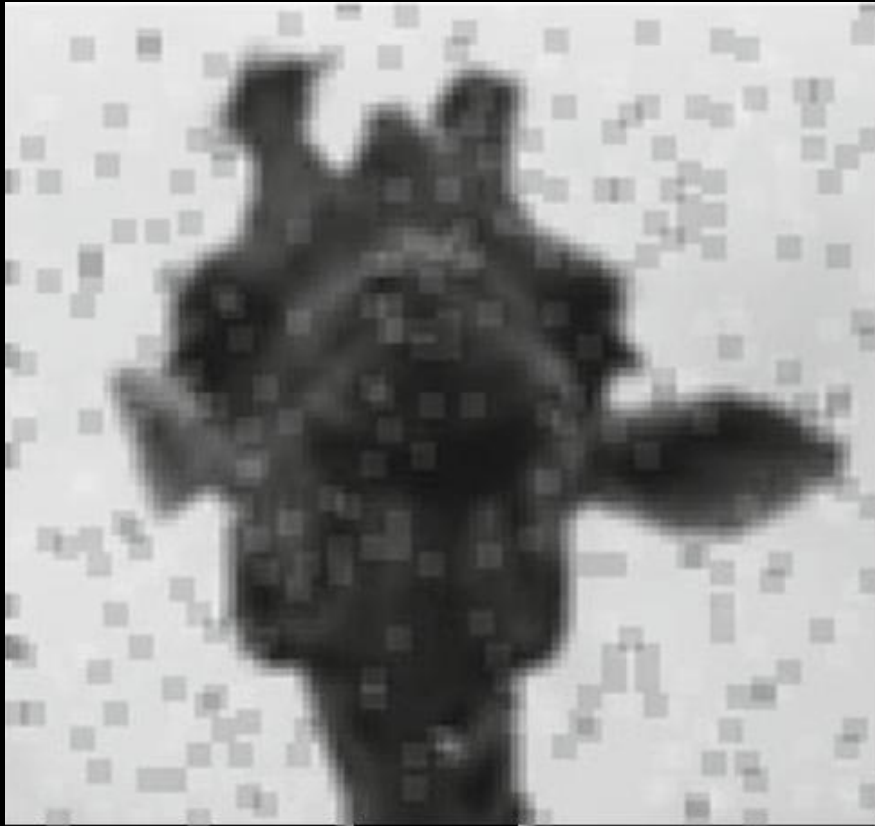


205	204	204	206	255
206	0	208	206	206
201	199	205	206	209
61	128	213	0	205
59	65	255	206	255

Ordering : [0, 199, 201, 204, 204, 205, 205, 206, 208]

Median = 204

Mean vs Median



Mean filtered



Median filtered

Yansıtıcı işaret yakalama

◆ connectedComponentsWithStats() [2/2]

```
int cv::connectedComponentsWithStats ( InputArray  image,  
                                     OutputArray labels,  
                                     OutputArray stats,  
                                     OutputArray centroids,  
                                     int          connectivity = 8 ,  
                                     int          ltype =  CV_32S  
                                     )
```

Python:

```
cv.connectedComponentsWithStats(          image[, labels[, stats[, centroids[, connectivity[, ltype]]]] ) -> retval, labels, stats, centroids  
cv.connectedComponentsWithStatsWithAlgorithm( image, connectivity, ltype, ccltype[, labels[, stats[, centroids]]] ) -> retval, labels, stats, centroids
```

```
#include <opencv2/imgproc.hpp>
```

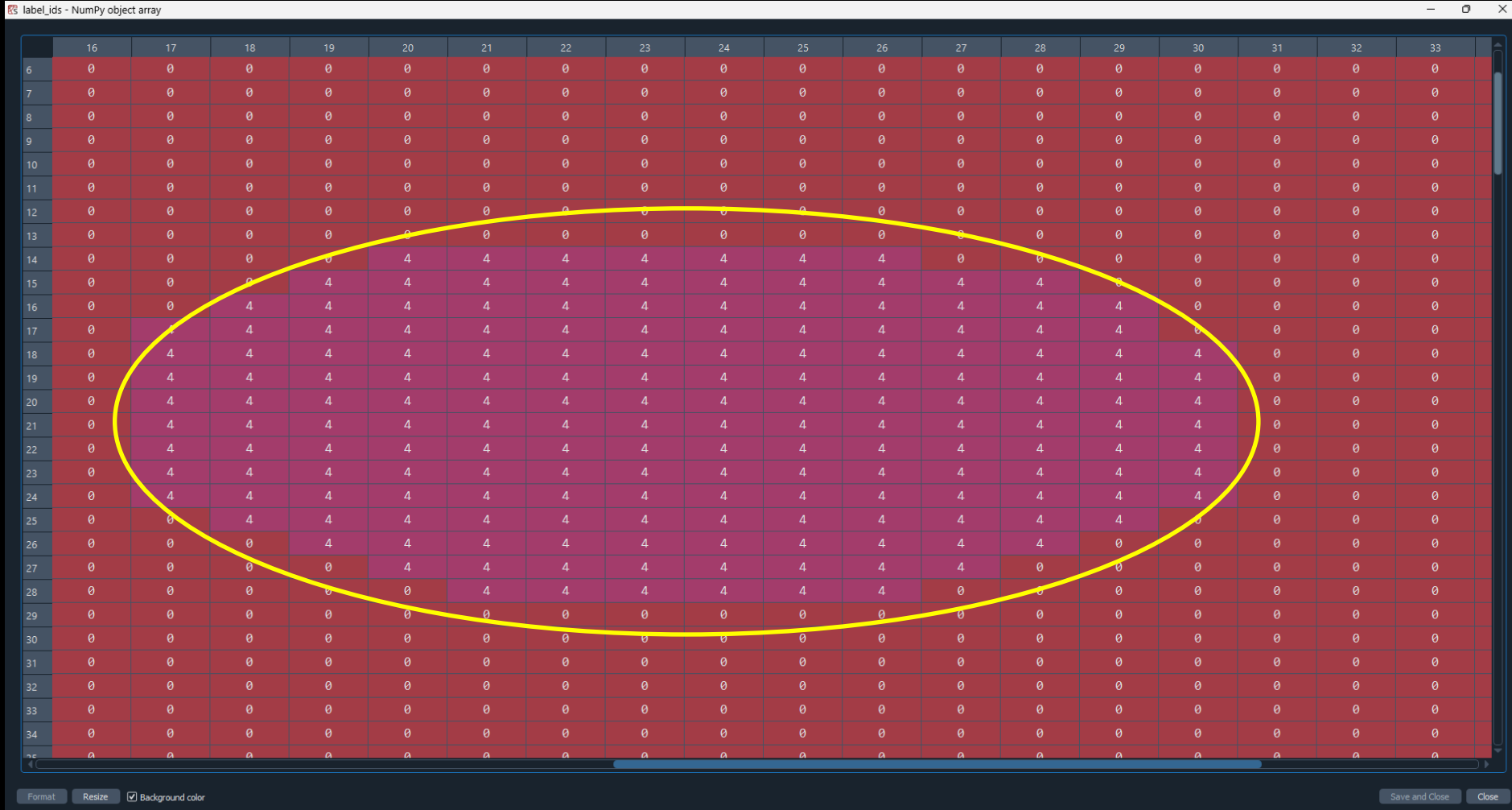
This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

- image** the 8-bit single-channel image to be labeled
- labels** destination labeled image
- stats** statistics output for each label, including the background label. Statistics are accessed via stats(label, COLUMN) where COLUMN is one of **ConnectedComponentsTypes**, selecting the statistic. The data type is CV_32S.
- centroids** centroid output for each label, including the background label. Centroids are accessed via centroids(label, 0) for x and centroids(label, 1) for y. The data type CV_64F.
- connectivity** 8 or 4 for 8-way or 4-way connectivity respectively
- ltype** output image label type. Currently CV_32S and CV_16U are supported.



Yansıtıcı işaret yakalama





Yansıtıcı işaret yakalama

◆ ConnectedComponentsTypes

enum `cv::ConnectedComponentsTypes`

```
#include <opencv2/imgproc.hpp>
```

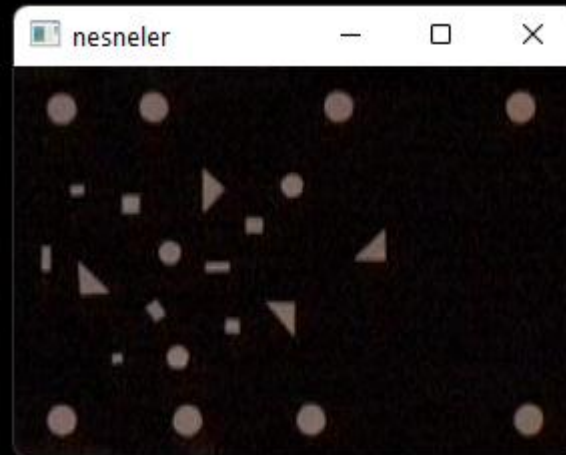
connected components statistics

Enumerator

CC_STAT_LEFT Python: <code>cv.CC_STAT_LEFT</code>	The leftmost (x) coordinate which is the inclusive start of the bounding box in the horizontal direction.
CC_STAT_TOP Python: <code>cv.CC_STAT_TOP</code>	The topmost (y) coordinate which is the inclusive start of the bounding box in the vertical direction.
CC_STAT_WIDTH Python: <code>cv.CC_STAT_WIDTH</code>	The horizontal size of the bounding box.
CC_STAT_HEIGHT Python: <code>cv.CC_STAT_HEIGHT</code>	The vertical size of the bounding box.
CC_STAT_AREA Python: <code>cv.CC_STAT_AREA</code>	The total area (in pixels) of the connected component.

Yansıtıcı işaret yakalama

```
BGR = cv.imread("nesneler.jpg")  
cv.imshow('nesneler',BGR)
```



```
imgray = cv.cvtColor(BGR, cv.COLOR_BGR2GRAY)
```



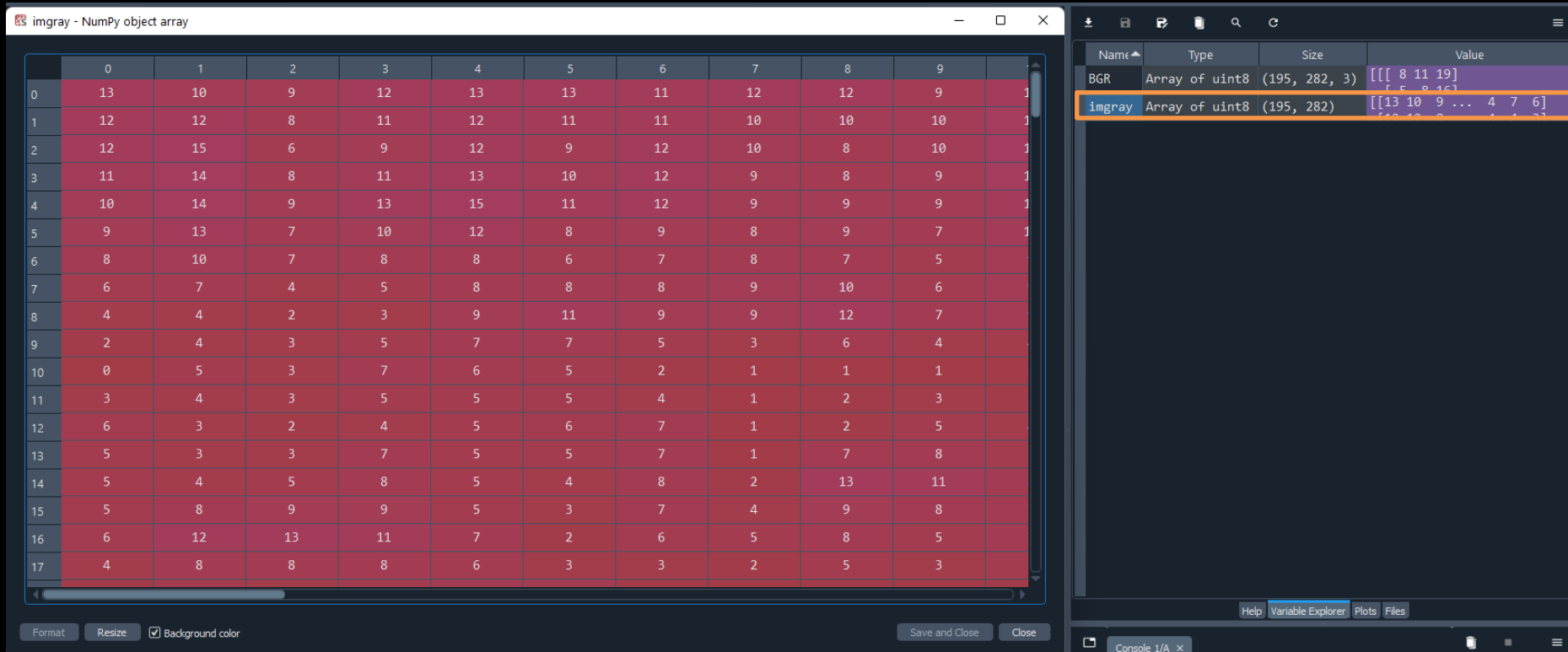


Yansıtıcı işaret yakalama

Name▲	Type	Size	Value
BGR	Array of uint8	(195, 282, 3)	<pre>[[[8 11 19] [5 8 16]</pre>
imgray	Array of uint8	(195, 282)	<pre>[[13 10 9 ... 4 7 6] [12 12 8 ... 4 4 3]</pre>

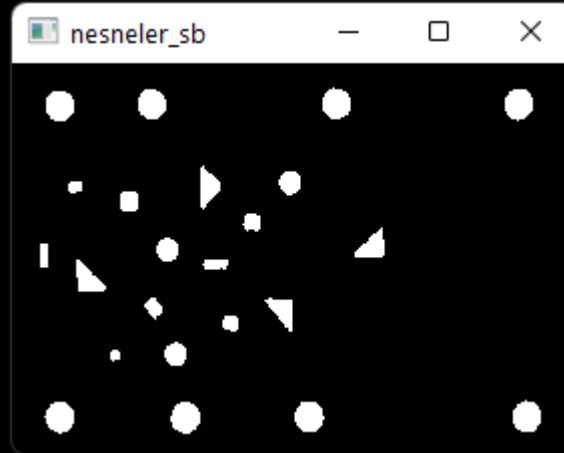


Yansıtıcı işaret yakalama

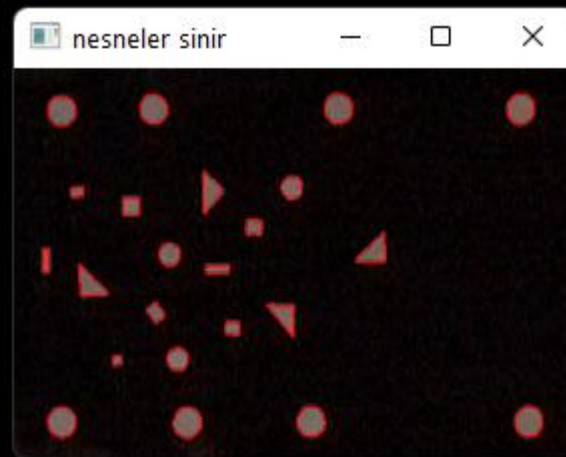


Yansıtıcı işaret yakalama

```
bw = cv.threshold(imgray, 127, 255, cv.THRESH_BINARY | cv.THRESH_OTSU)[1]
```

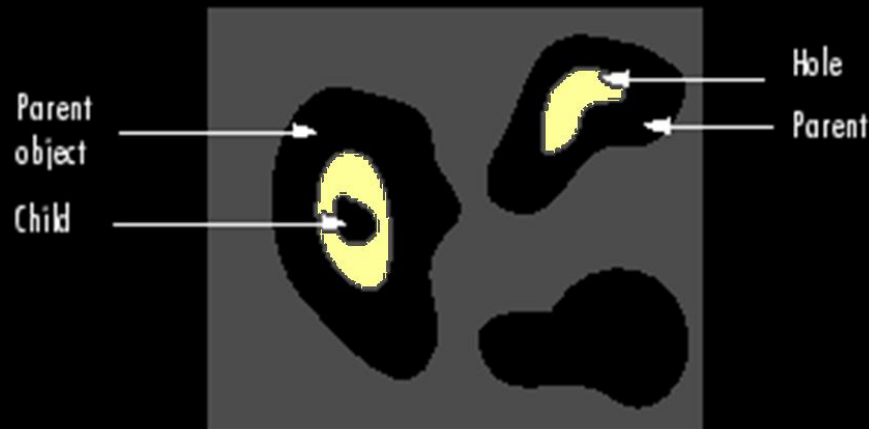


```
contours, hierarchy = cv.findContours(bw, cv.RETR_EXTERNAL, cv.CHAIN_APPROX_NONE)  
cv.drawContours(BGR, contours, -1, (0, 0, 255), 1)
```



Yansıtıcı işaret yakalama

`cv.RETR_EXTERNAL`, it returns only extreme outer flags. All child contours are left behind.



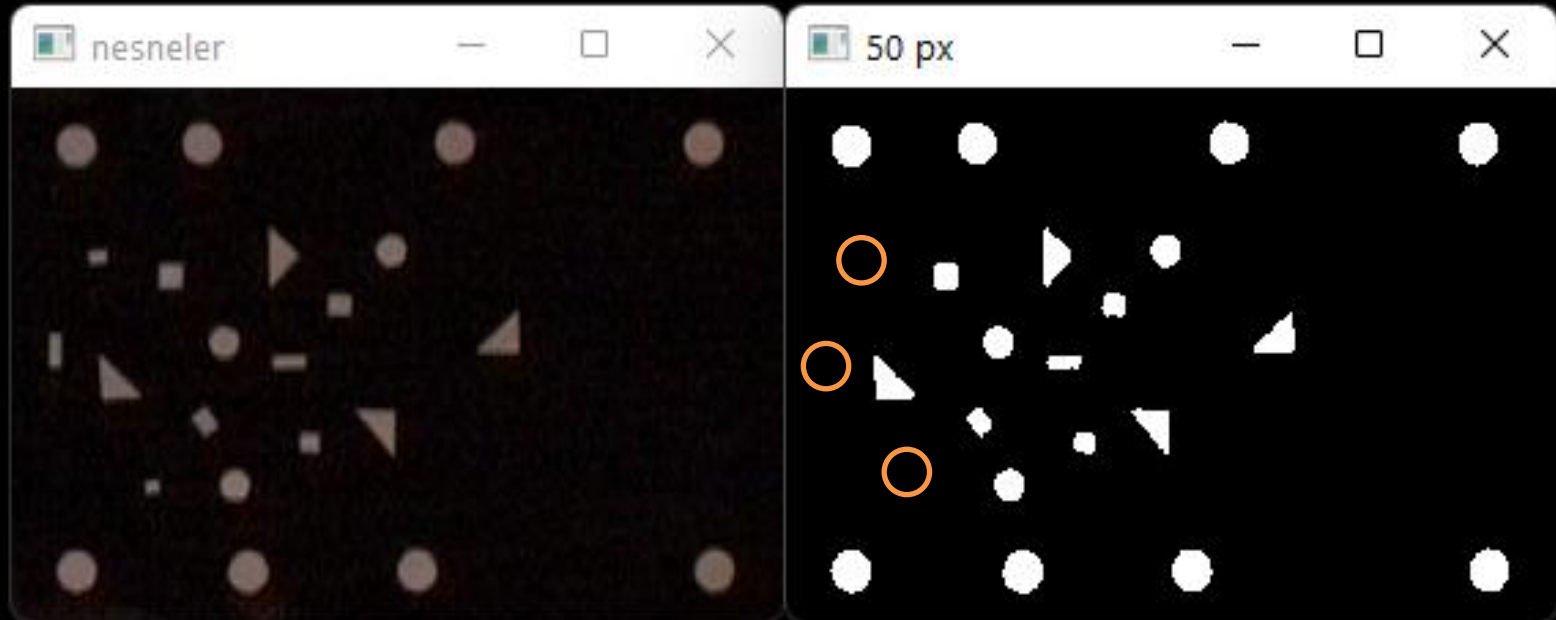
```
## Contour Tracing
```

```
# cv.CHAIN_APPROX_SIMPLE for circle returns only one point
```

```
# cv.CHAIN_APPROX_NONE returns all contours
```


Yansıtıcı işaret yakalama

```
# Alanı 50 pikselden az olan  
# nesneleri resimden sil
```



Yansıtıcı işaret yakalama

```
import cv2 as cv
import numpy as np

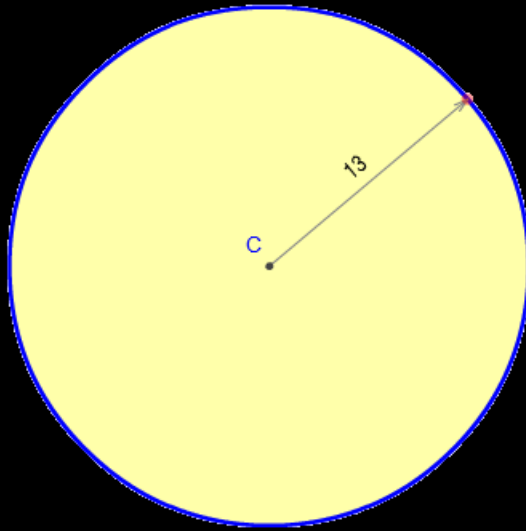
BGR = cv.imread("nesneler.jpg")
cv.imshow('nesneler',BGR)
imgray = cv.cvtColor(BGR, cv.COLOR_BGR2GRAY) # Convert to Gray

# Convert to the Black & White binary image
bw = cv.threshold(imgray, 127, 255, cv.THRESH_BINARY | cv.THRESH_OTSU)[1]
nb_components, output, stats, centroids =
cv.connectedComponentsWithStats(bw, connectivity=8)
sizes = stats[1:, -1]
nb_components = nb_components - 1

min_blob_size = 50
# remove blobs
mask_clean = np.zeros((output.shape))
# for every component in the image, keep it only if it's above min_blob_size
for i in range(0, nb_components):
    if sizes[i] >= min_blob_size:
        mask_clean[output == i + 1] = 255

cv.imshow('50 px',mask_clean)
cv.waitKey(5)
```

Dairenin Geometrik Özellikleri



$$\text{Çevre} = 2.\pi.r = 2 \times 3.1415 \times 13 = 81.7$$

$$\text{Çevre} = \pi.d = 3.1415 \times 26 = 81.7$$

$$\text{Çevre} = \sqrt{4.\pi. \text{Alan}} = \sqrt{4 \times 3.1415 \times 530.9} = 81.7$$

$$\text{Çevre} = \sqrt{4.\pi. \text{Alan}} = \sqrt{4 \times 3.1415 \times 530.9} = 81.7$$

$$\text{Alan} = \pi.r^2 = 3.1415 \times (13)^2 = 530.9$$

$$\text{Alan} = (\text{Çevre})^2 / 4.\pi = (81.7)^2 / 4 \times 3.1415 = 530.9$$

$$\text{Yuvarlaklık Katsayısı} = 4.\pi.\text{Alan} / (\text{Çevre})^2 = 1$$

Yuvarlaklık Katsayısı $\approx 1 \rightarrow$ Daha Yuvarlak

Yansıtıcı işaret yakalama

```
import cv2 as cv
import numpy as np

BGR = cv.imread("nesneler.jpg")
# Convert to the gray
imggray = cv.cvtColor(BGR, cv.COLOR_BGR2GRAY) # Convert to Gray
# Convert to the Black & White binary image
bw = cv.threshold(imggray, 127, 255, cv.THRESH_BINARY | cv.THRESH_OTSU)[1]
# median filter salt&pepper
filtered = cv.medianBlur(bw, 3)
# getting labelling with connectComponents
totalLabels, label_ids, values, centroid = cv.connectedComponentsWithStats(filtered)
contours, hierarchy = cv.findContours(filtered, cv.RETR_EXTERNAL, cv.CHAIN_APPROX_NONE)

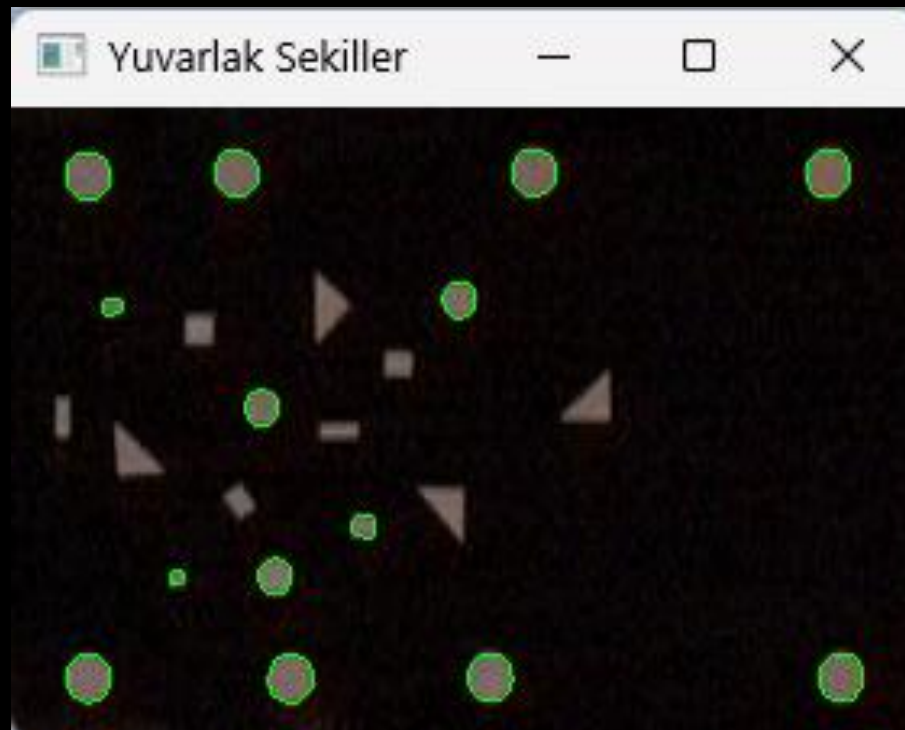
for i, c in enumerate(contours):
    area = cv.contourArea(c)
    perimeter = cv.arcLength(c, True)
    ratio = (4*np.pi*area) / (perimeter**2)
    print(f"{i}. object's area : {area:.1f} and perimeter : {perimeter:.3f}: {ratio}")
    if ratio >= 0.879:
        cv.drawContours(BGR, c, -1, (0, 255, 0), 1)

cv.imshow('Yuvarlak Sekiller', BGR)
cv.waitKey(0)

cv.destroyAllWindows()
```


Yansıtıcı işaret yakalama

```
if ratio >= 0.879:  
    cv.drawContours(BGR, c, -1, (0, 255, 0), 1)
```



0.879 ???

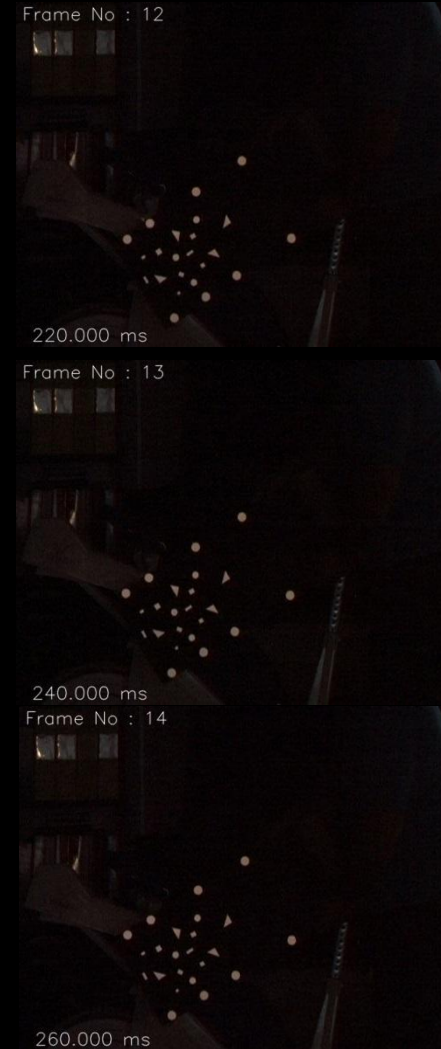


Yansıtıcı işaret yakalama

0. object's area :	155.5	and perimeter :	46.385:	0.90822
1. object's area :	157.0	and perimeter :	46.627:	0.90746
2. object's area :	161.5	and perimeter :	48.042:	0.87932
3. object's area :	156.0	and perimeter :	46.627:	0.90168
4. object's area :	14.5	and perimeter :	14.243:	0.89825
5. object's area :	87.0	and perimeter :	34.971:	0.89397
6. object's area :	42.0	and perimeter :	24.485:	0.88034
7. object's area :	102.5	and perimeter :	46.385:	0.59866
8. object's area :	47.5	and perimeter :	26.385:	0.85743
9. object's area :	40.5	and perimeter :	27.071:	0.69447
10. object's area :	117.5	and perimeter :	48.042:	0.63975
11. object's area :	31.0	and perimeter :	25.657:	0.59179
12. object's area :	88.0	and perimeter :	34.971:	0.90425
13. object's area :	111.5	and perimeter :	46.385:	0.65123
14. object's area :	48.5	and perimeter :	27.071:	0.83165
15. object's area :	70.0	and perimeter :	31.657:	0.87775
16. object's area :	25.5	and perimeter :	19.071:	0.88105
17. object's area :	88.0	and perimeter :	34.971:	0.90425
18. object's area :	112.0	and perimeter :	47.799:	0.61601
19. object's area :	155.0	and perimeter :	45.799:	0.92860
20. object's area :	155.0	and perimeter :	45.799:	0.92860
21. object's area :	153.0	and perimeter :	45.799:	0.91662
22. object's area :	150.5	and perimeter :	45.213:	0.92516

0.879

Yansıtıcı işaret yakalama



Yansıtıcı işaret yakalama

```
import cv2 as cv
import numpy as np
import os
from sys import exit

fileName = 'C:\\Lectures\\...\\nesneler.avi'
# Get the base name for saving the image
base = os.path.splitext(os.path.basename(fileName))[0]
# Set the font for writing on the image
font = cv.FONT_HERSHEY_SIMPLEX
# Capture the image from video file
cap = cv.VideoCapture(fileName)
# Check if camera opened successfully
if not cap.isOpened():
    print("Error opening video stream or file")
    exit()
else:
    frame_width = int(cap.get(cv.CAP_PROP_FRAME_WIDTH)) # cap.get(3)
    frame_height = int(cap.get(cv.CAP_PROP_FRAME_HEIGHT)) # 4
    length = int(cap.get(cv.CAP_PROP_FRAME_COUNT)) # 7

    print(f"Video frame width {frame_width} and height {frame_height}")
    print(f"Numbers of video frame is {length}")
    print(f"Frame rate {cap.get(cv.CAP_PROP_FPS):.3f} Hz") # cap.get(5)
```


Yansıtıcı işaret yakalama

```
# Read until video is completed
while(cap.isOpened()):
    # Capture frame-by-frame
    ret, frame = cap.read()
    # if frame is read correctly ret is True
    if not ret:
        print("Can't receive frame (stream end?). Exiting ...")
        break
    # Display the resulting frame
    mSec = cap.get(cv.CAP_PROP_POS_MSEC) # cap.get(0)
    frameNo = int(cap.get(cv.CAP_PROP_POS_FRAMES))
    cv.putText(frame, f'Frame No:{frameNo}', (10, 30), font, 1, (255,255,255), 1, cv.LINE_AA)
    cv.putText(frame, f'{mSec:>8.3f} ms', (10, frame_height-10), font, 1, (255,255,255), 1, cv.LINE_AA)
    cv.imwrite(base+'_'+(str(frameNo)).zfill(5)+'.jpg', frame, [int(cv.IMWRITE_JPEG_QUALITY), 100])
    cv.imshow('Frame', frame)

    # Press Q on keyboard to exit
    if cv.waitKey(1) == ord('q'):
        print("'q' pressed to exit")
        break

# When everything done, release the video capture object
cap.release()
# Closes all the frames
cv.destroyAllWindows()
```



Yansıtıcı işaret yakalama



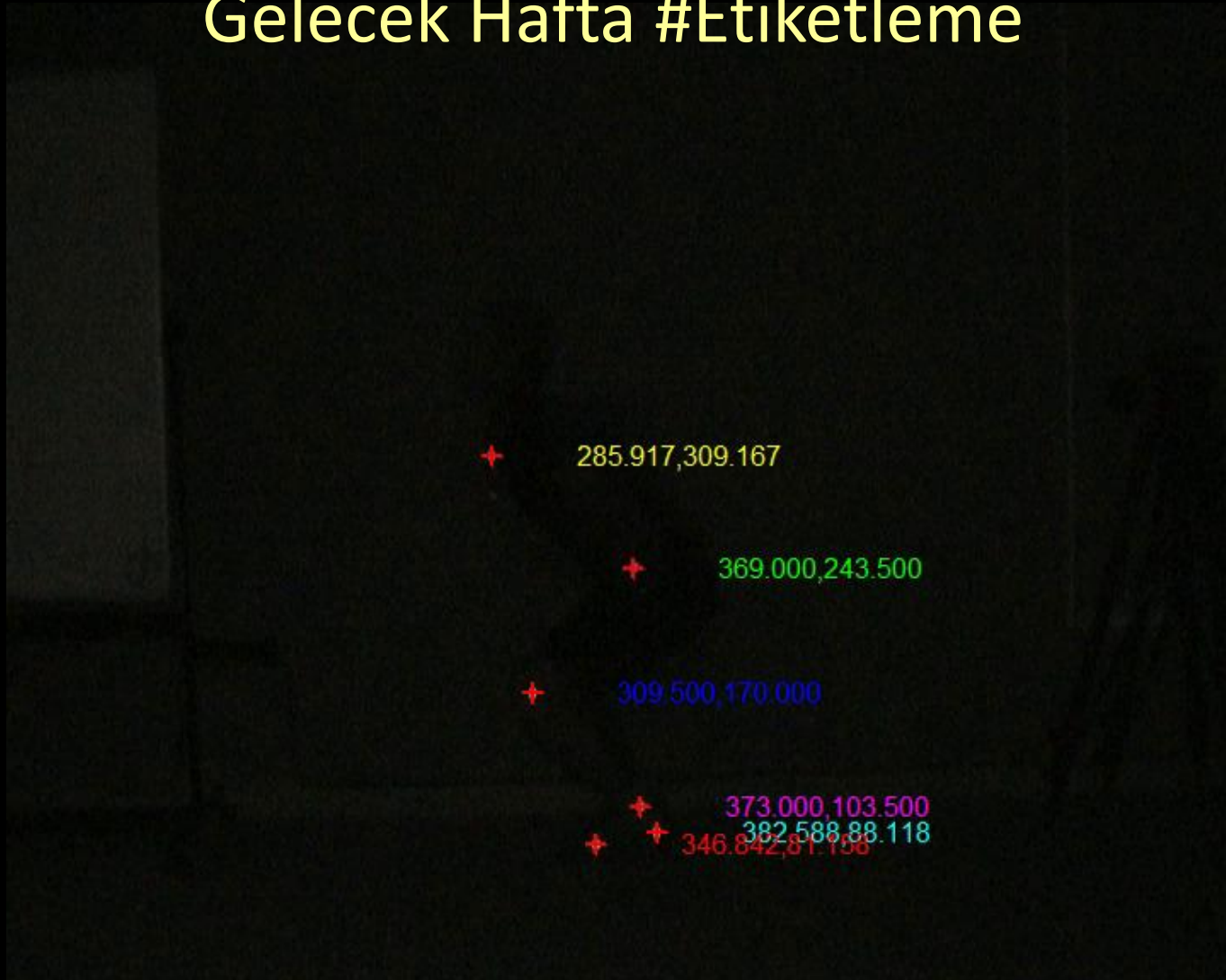


Homework #2





Gelecek Hafta #Etiketleme





Gelecek Hafta #Etiketleme

