Gİ-9
NESNE ÖZELLİKLERİ

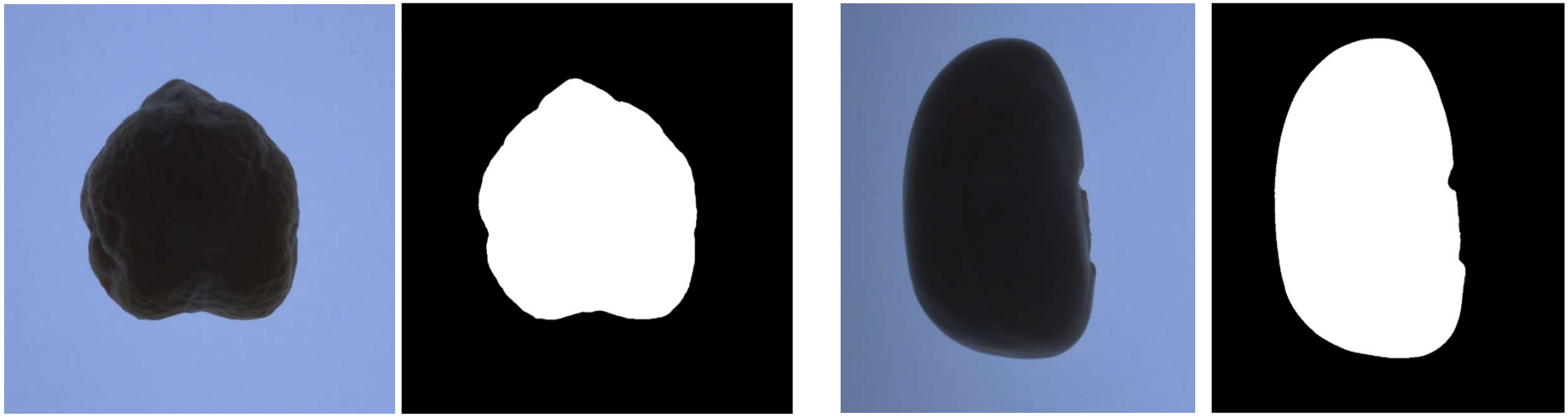
Dr. Öğr. Üyesi Muhammet Üsame ÖZİÇ
PAÜ Biyomedikal Mühendisliği

SUNUM PLANI

- 1-) Segmentasyon
- 2-)Kontur (Contour Kavramı)
- 3-) Çevre Çizimi
- 4-) Sınırlayıcı Kutu (Bounding Box)
- 5-) Alan
- 6-) Nesne Momentleri
- 7-) Ağırlık merkezi
- 8-) Nesne Çap ve Yarıçap
- 9-) Aspect Ratio (En-Boy Oranı)
- 10-) Extent
- 11-) Solidity
- 12-) Ellipse
- 13-) Eccentricity
- 14-) Çoklu Nesne Özellikleri
- 15-) Skimage yaklaşımı

1-) Segmentasyon

- Nesne özelliklerinin elde edilmesi için ilgili nesnenin binary formatına dönüştürülmesi gerekir.
- Yani nesne ön plana (foreground) geriye kalan her şey arka plan kabul (background)edilmelidir.



- Görüntü özelliklerinden dolayı her zaman iyi bir segmentasyon işlemi yapılamayabilir.
- Arka plan karmaşıklığı, renk farklılıkları, nesnelerin yapışık olması, ortam ışığının uygun açıda olmaması, nesne üzerinde farklı tonların olması vb.
- Bundan dolayı farklı görüntü ön işleme yöntemleri ve konvansiyonel görüntü işleme metotları uygulanabilir.
- Eğer nesne ölçümünün gerçekleştirileceği ortam kişi tarafından ayarlanabilirse arka plan ve ön plan düzgün bir şekilde ayarlanabilir.
- Işığın geliş açısı ve arka plan ışığı ayarlanabilir.
- Böylece kullanılacak ön işleme yöntemleri görüntünün özelliklerine göre oldukça azaltılabilir.

Kullanılabilecek Ön ve Son İşleme Metotları

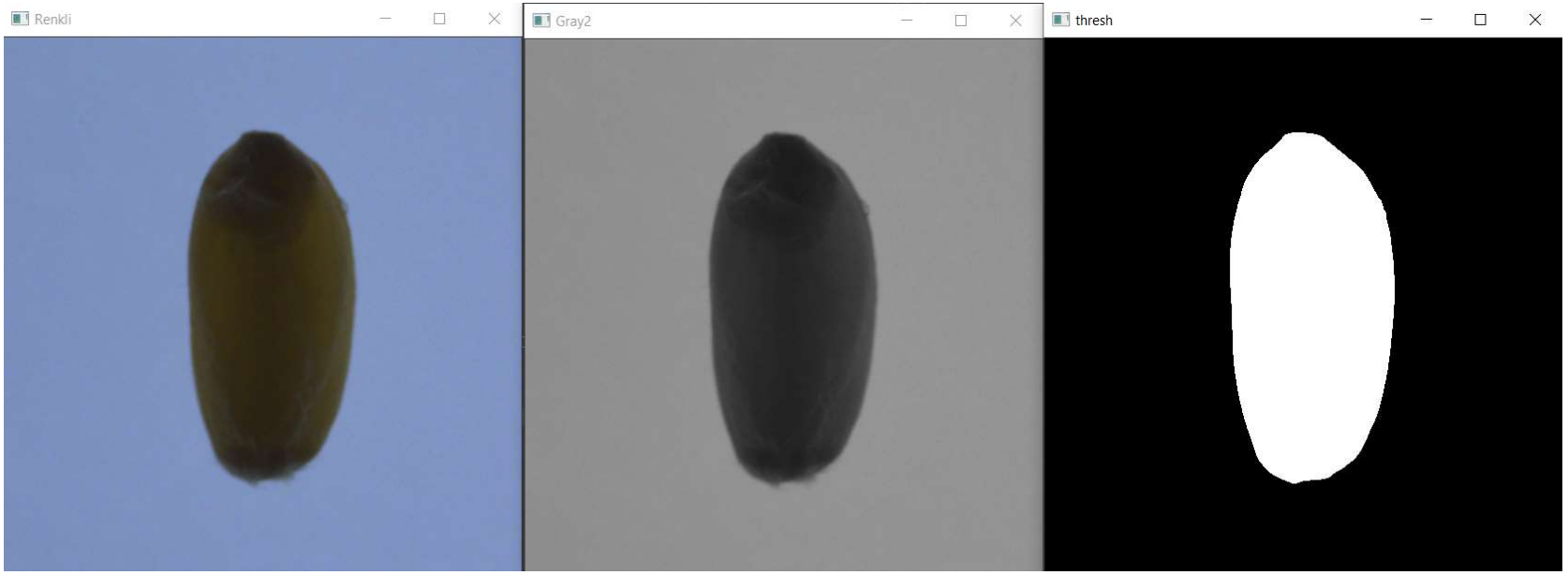
- Threshold
- Işık ve Gölge Giderme
- Morfolojik Operatörler
- Çizim ve Bilgi Gösterme Metotları
- Fourier Dönüşümü
- Gürültü Giderme
- Kenar Keskinleştirme
- Kenar Bulma

- Yapay Zeka
- Deep learning
- Birçok hazır algoritmalar
- Segmentasyon Algoritmaları

(k-means, fuzzy c means, OTSU, Watershed vb.)

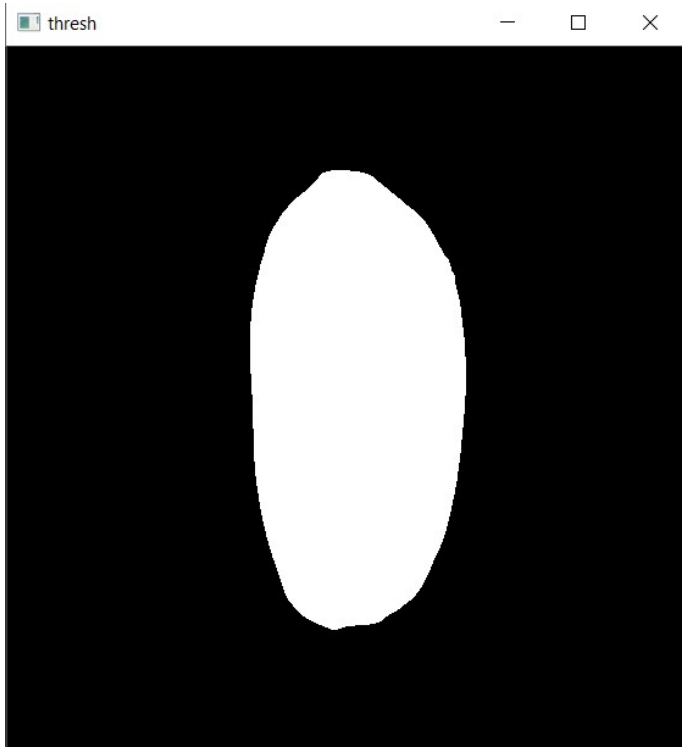
Görüntü renkli ise gri seviyeye dönüştürülmelidir.

```python
import numpy as np
import cv2
from scipy.spatial import distance as dist
import math
import math, os
import imutils

img = cv2.imread('R1_BUGDAY_TEK_renkli.png')
img15 = cv2.imread('R1_BUGDAY_TEK_renkli.png')

gray2 = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
gray=cv2.blur(gray2,(5,5))
_, thresh = cv2.threshold(gray, 50, 255, cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)

cv2.imshow("Renkli",img15)
cv2.imshow("Gray2",gray2)
cv2.imshow("thresh",thresh)
```



2-)Kontur (Contour Kavramı)

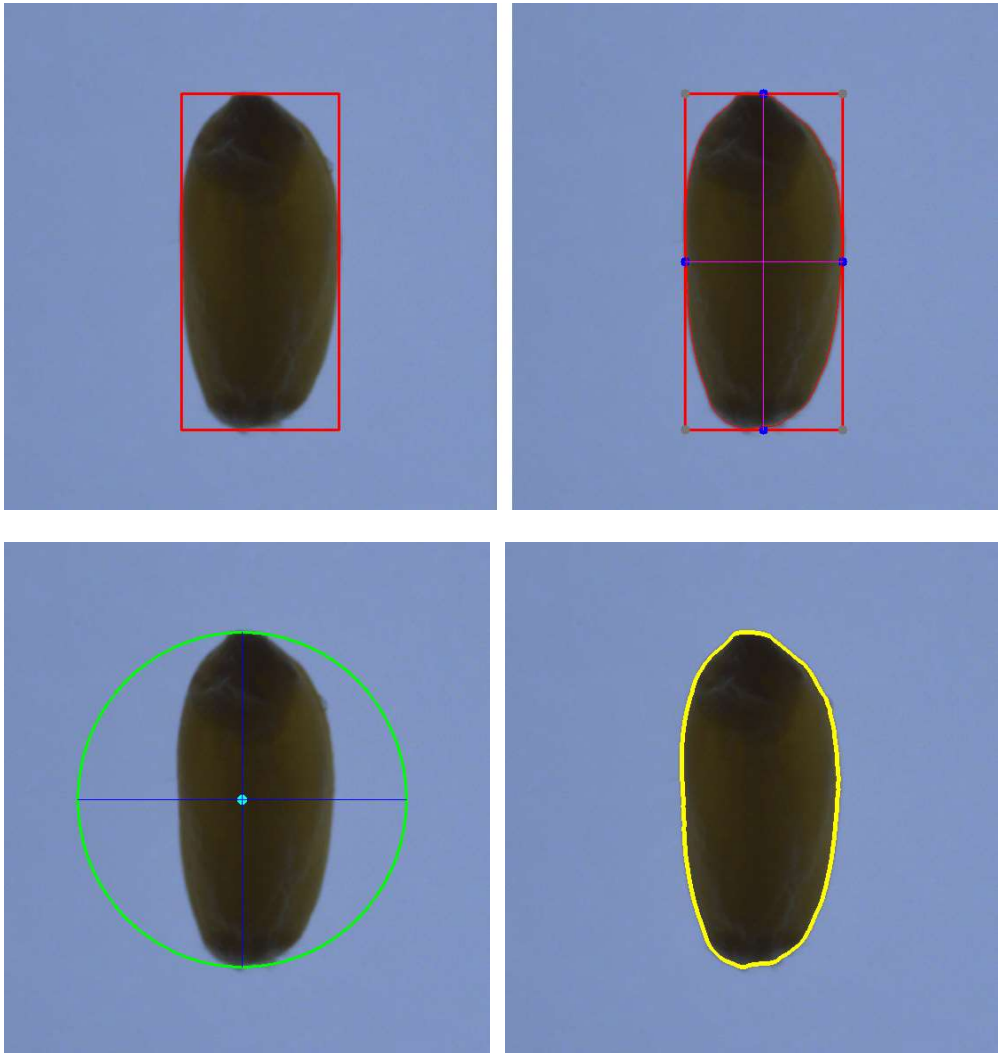
- Her bir nesne yapı olarak düzenli veya düzensiz bir geometrik şekildir.
- Kontur sınır anlamına da gelmektedir.
- Geometrik nesnenin konturlarının bulunması demek onu sınırlayan çevre çizgilerinin bulunması anlamında gelir.
- Yandaki şekilde siyah çizgi ile gösterilen noktalar bütünü üçgenin konturlarını temsil etmektedir.
- Geometrik şeklin sınırları boyunca ard arda devam eder ve benzer renk özelliğine sahiptir.



- OpenCV kontur bulma için pratik bir fonksiyon sunar
- Nesnenin sınırları segmentasyon veya threshold ile net bir şekilde belirlenebilirse kontur özelliği ile birçok geometrik özellik elde edilebilir.

```
_, thresh = cv2.threshold(gray, 50, 255, cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)

cnts,_ = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)[-2]

for contour in cnts:
```

- Fonksiyonun döndürdüğü ‘cnts’ parametesi içinde koordinatları ve diğer bazı görüntü özelliklerinin bulunduğu bir sözlüktür.
- print ile cnts sözlüğünün tuttuğu değerler incelenebilir.
- Cnts içerisinde bilgisi tutulan birden fazla nesne var ise for döngüsü ile tek tek hesaplamaları gerçekleştirilir



- Literatürde Kontur Özellikleri (Contour Features) olarak ifade edilir
- Alan
- Çevre
- Geometri Merkezi
- Çevreleyici Geometriler
- Ve birçok başka özellik…

3-Çevre Çizimi

- Orjinal görüntüdeki nesnenin çevresinin çizdirilmesi
- Perimeter


```
img_cevre=img.copy()
cv2.drawContours(img_cevre, contour, 0, (0, 255, 255), 3)
```

4-) Çevre Değeri

- Çevreyi sınırlayan piksellerin toplam değerini verir


```
rounded_perimeter = round(perimeter, 2)
print("Çevre Miktarı:", rounded_perimeter)
```

```
Çevre Miktarı: 953.45
```

```
# ÇEVRE ÇİZİMİ
img_cevre=img.copy()
cv2.drawContours(img_cevre, contour, 0, (0, 255, 255), 3)
perimeter = cv2.arcLength(contour,True)
print(perimeter)
```

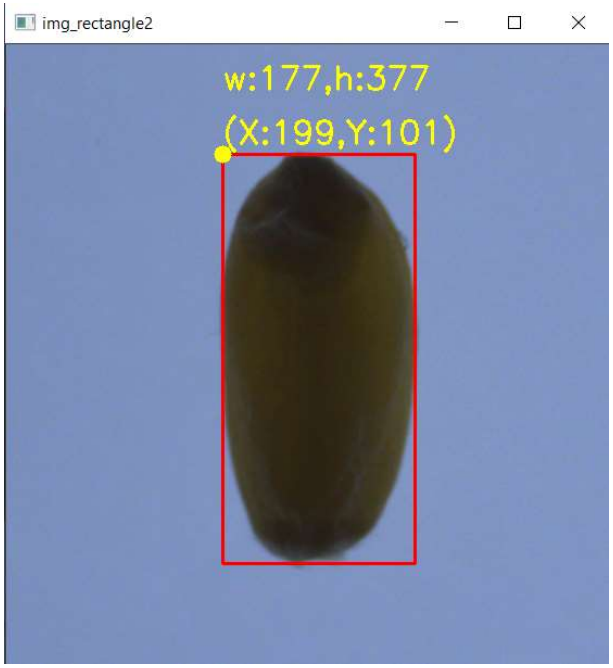
```
953.4528790712357
```

4-Sınırlayıcı Kutu (Bounding Box)

- Nesnenin içine sığabileceği en küçük sınırlayıcı kutu başlangıç koordinat değerlerini (x,y)
- w= (x,y) noktasından itibaren sınırlayıcı kutunun genişliğini hesaplar
- h= (x,y) noktasından itibaren sınırlayıcı kutunun yükseklik değerini hesaplar


```
img_rectangle=img.copy()
(x, y, w, h) = cv2.boundingRect(contour)
cv2.rectangle(img_rectangle, (x, y), (x + w, y + h), (0, 0, 255), 2)
```

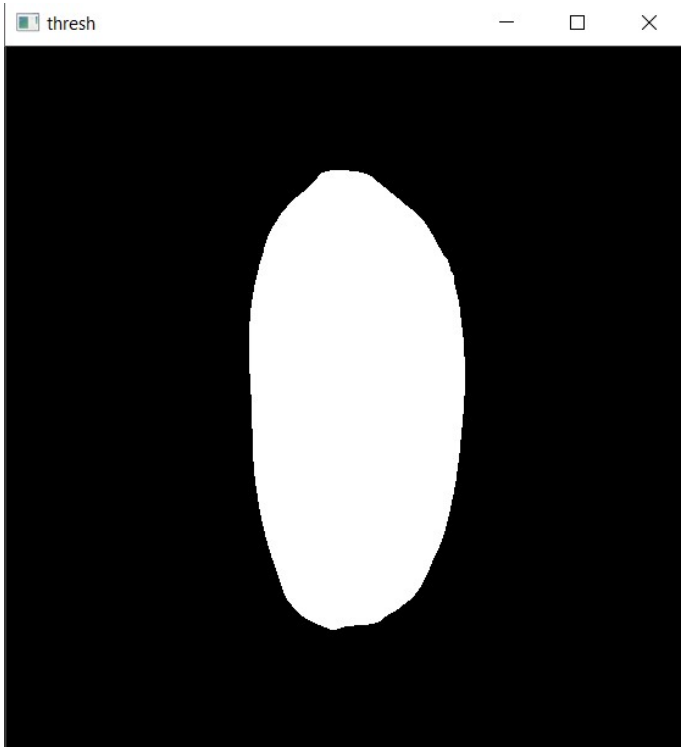
Nesne sayma işlemi yapılırken nesne etrafına sınırlayıcı kutu çizdirilebilir
w ve h piksel cinsinden miktarı ifade eder




```
img_rectangle2=img.copy()
(x, y, w, h) = cv2.boundingRect(contour)
cv2.rectangle(img_rectangle2, (x, y), (x + w, y + h), (0, 0, 255), 2)
cv2.circle(img_rectangle2, (x,y), 8, (0, 255, 255), -1)
cv2.putText(img_rectangle2, "(X:"+str(x)+","+"Y:"+str(y)+")" ,(x,y-10),cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,255),2)
cv2.putText(img_rectangle2, "w:"+str(w)+","+"h:"+str(h)+"" ,(x,y-60),cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,255),2)
```

5-) Alan

- Piksel cinsinden alanı verir
- Eğer bir pikselin gerçek dünyada en ve boy uzunluğu bilinirse
- Ve bir pikselin mm2 cinsinden değeri bulunabilirse
- Nesnenin gerçek dünyada kapladığı alan
- Toplam piksel sayısı X Bir pikselin kapladığı alanın gerçek dünyadaki değeri
- Formülü ile bulunabilir




```
#ALAN BULMA
area = cv2.contourArea(contour)
print(area)
```

```
53441.5
```

6-) Nesne Momentleri

- Nesne momentleri, bir görüntünün şeklini ve dağılımını tanımlayan istatistiksel özelliklerdir.
- Bu özellikler, bir nesnenin konumunu, büyüklüğünü, yönelimini ve şeklini temsil eder.
- Nesne momentleri, bir görüntünün pikselleri arasındaki ilişkileri hesaplayarak elde edilir.
- Bu hesaplamalar, nesnenin yoğunluk dağılımı üzerinde yapılan matematiksel operasyonlara dayanır.
- Nesne momentleri, nesnenin merkezi, yarıçapı, yönü, açısı ve asimetrisi gibi bilgileri ifade edebilir.
- Moment tabanlı özellikler, nesnelerin benzersiz özelliklerini temsil edebilir ve desen tanıma algoritmalarında kullanılarak nesne tespiti ve sınıflandırma performansını artırabilir.

- Python'da **sözlük** (*dictionary*), **anahtar-değer (key-value)** çiftlerini depolamak için kullanılan bir veri yapısıdır.
- **Anahtarlar benzersizdir (unique):** Her anahtar yalnızca bir kez kullanılabilir.
- **Anahtarlar değiştirilemez (immutable):** Anahtarlar, yalnızca değiştirilemez veri tipleri (örneğin, str, int, tuple) olabilir.
- **Değerler değiştirilebilir (mutable):** Değerler herhangi bir veri tipi olabilir ve değiştirilebilir

Boş bir sözlük oluşturma:

python

```python
sozluk = {}
# veya
sozluk = dict()
```

Anahtar-değer çiftleriyle sözlük oluşturma:

python

```python
sozluk = {
    "ad": "Ahmet",
    "yas": 25,
    "sehir": "İstanbul"
}
```

- CV2.MOMENTS komutu ile contour değişkeni içindeki nesne değerlerinin benzersiz özelliklerini otomatik olarak hesaplanır.
- Bir sözlük olarak tutar. (Python Bilgisinden)
- Aşağıdaki key value değerleri otomatik hesaplanır
- Bu değerlerin farklı kombinasyonları ile nesne özellikleri elde edilebilir.

```
# AĞIRLIK MERKEZİ BULMA
img_centroid=img.copy()
M = cv2.moments(contour)
```

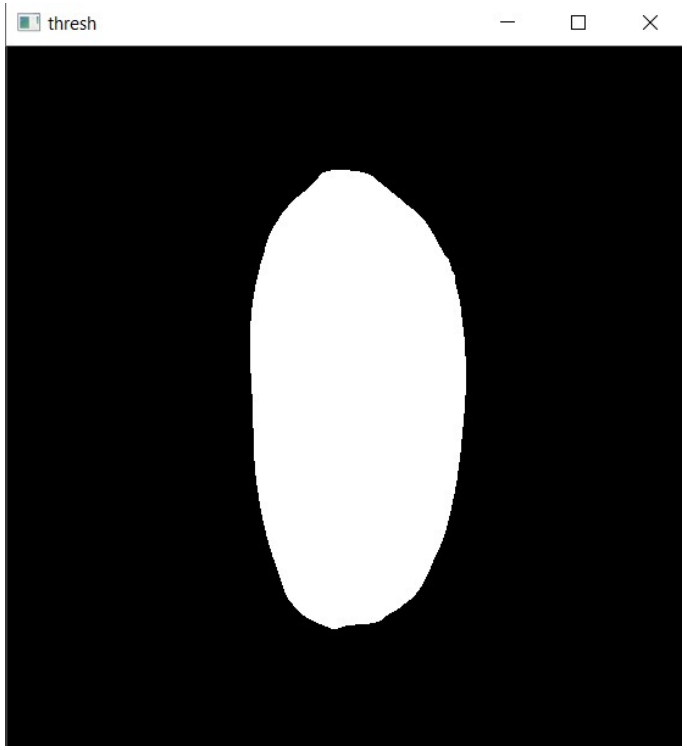
{'m00': 15946170.0, 'm10': 2512158510.0, 'm01': 2140943280.0, 'm20': 568586636580.0, 'm11': 337099003380.0, 'm02': 422375441400.0, 'm30': 144203744893170.0, 'm21': 76698286353210.0, 'm12': 66530197502520.0, 'm03': 95616562341210.0, 'mu20': 172821360067.4945, 'mu11': -185051680.2001953, 'mu02': 134930987454.18365, 'mu30': 176115778254.9375, 'mu21': 417776338746.1203, 'mu12': 38890234024.70713, 'mu03': 2676311195639.547, 'nu20': 0.0006796489325593325, 'nu11': -7.277467141055643e-07, 'nu02': 0.0005306386985763648, 'nu30': 1.7344320571703332e-07, 'nu21': 4.114365457929711e-07, 'nu12': 3.83000712779222e-08, 'nu03': 2.6356979361393227e-06}

7-) Ağırlık Merkezi (Centroid)

- Nesnenin ağırlık merkezi momentlerin farklı kombinasyonları ile x,y değerleri bulunur.
- Ağırlık merkezini göstermek için bir daire atılır.

```
# AĞIRLIK MERKEZİ BULMA
img_centroid=img.copy()
M = cv2.moments(contour)
cX = int(M["m10"] / M["m00"])
cY = int(M["m01"] / M["m00"])
Centroid = (int(cX),int(cY))
cv2.circle(img_centroid, (cX, cY), 5, (0, 255, 255), -1)
```


```
# AĞIRLIK MERKEZİ BULMA
img_centroid=img.copy()
M = cv2.moments(contour)
cX = int(M["m10"] / M["m00"])
cY = int(M["m01"] / M["m00"])
Centroid = (int(cX),int(cY))
cv2.circle(img_centroid, (cX, cY), 5, (0, 255, 255), -1)
cv2.putText(img_centroid,"(cX:"+str(cX)+","+"cY:"+str(cY)+")" ,(x,y-10),cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,255),2)
```



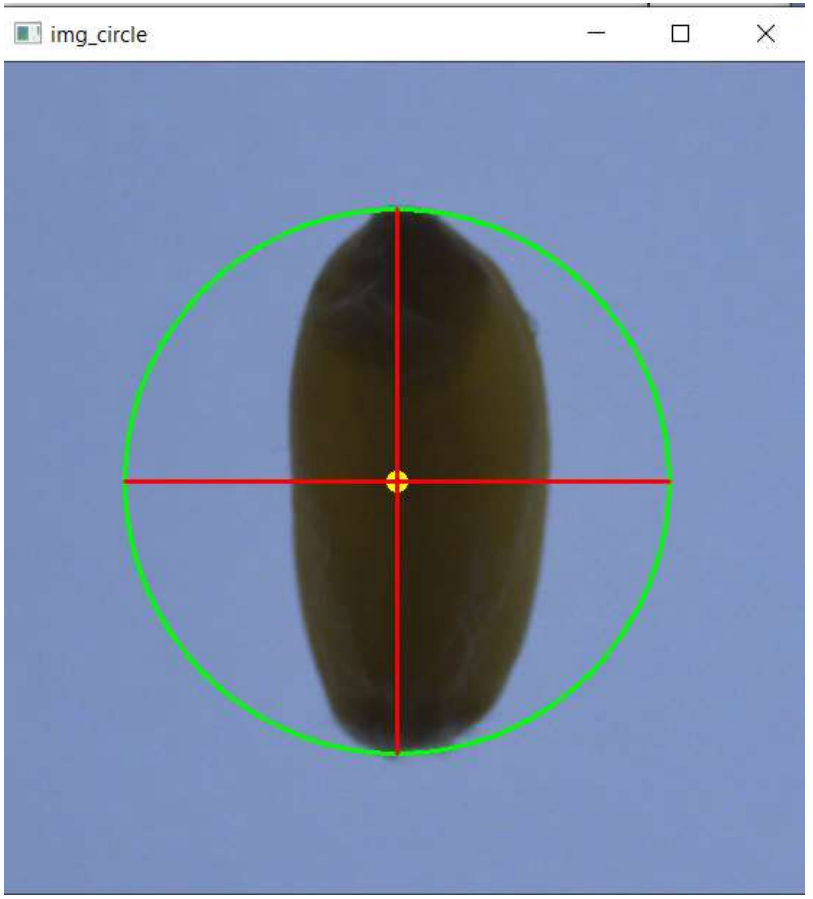

```
# MOMENT İLE ALAN
print(M['m00'])
```

```
53441.5
```

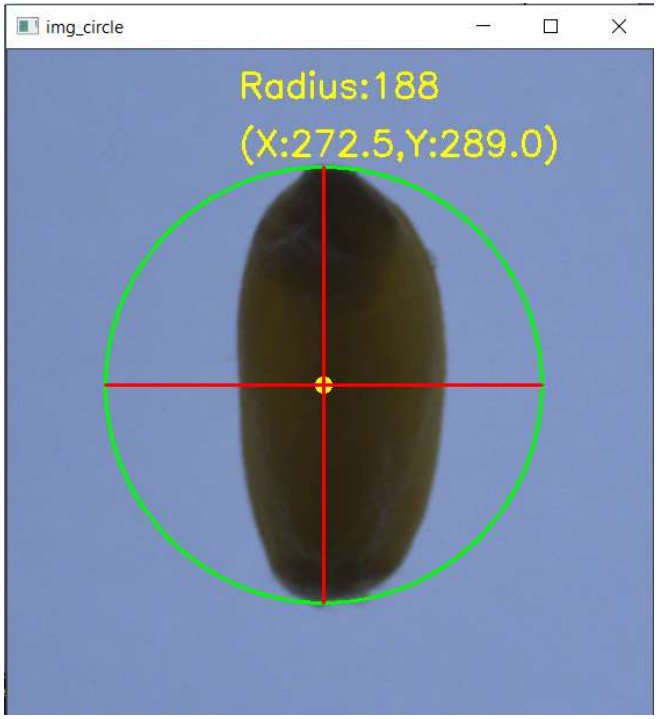
Moment kullanılarak alan hesabı piksel cinsinden yapılabilir

8-) Nesne Çap ve Yarıçap

- Nesneyi çevreleyen ve nesnenin içine sığdığı en küçük daire bulunarak nesnenin çap ve yarıçapı bulunabilir.




```
 # 5-Circle Çizdir
img_circle=img.copy()
(l, k), radius = cv2.minEnclosingCircle(contour)
center = (int(l), int(k))
radius = int(radius)
cv2.circle(img_circle, center, radius, (0, 255, 0), 2)
cv2.circle(img_circle,center,5,(0,255,255),3)
cv2.line(img_circle, center, (center[0], center[1]+radius), (0,0,255), 2)
cv2.line(img_circle, center, (center[0], center[1]-radius), (0,0,255), 2)
cv2.line(img_circle, center, (center[0]+radius, center[1]), (0,0,255), 2)
cv2.line(img_circle, center, (center[0]-radius, center[1]), (0,0,255), 2)
```



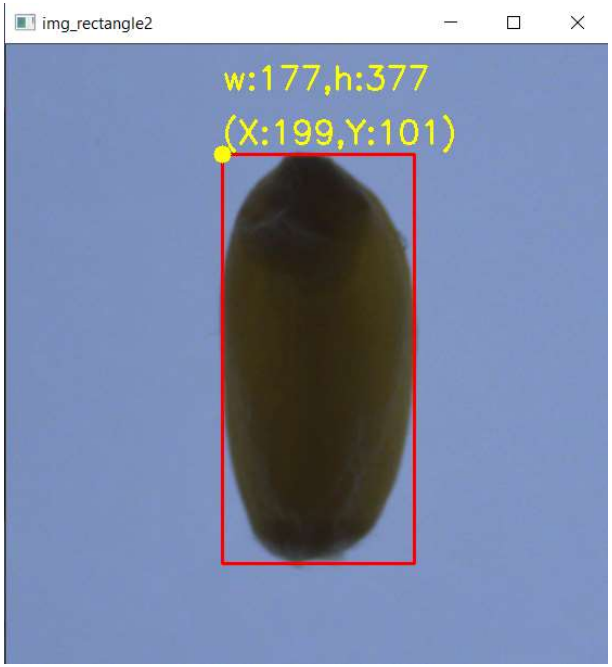
Radius: Yarıçap
Diameter: Çap

```
 # 5-Circle Çizdir
img_circle=img.copy()
(l, k), radius = cv2.minEnclosingCircle(contour)
center = (int(l), int(k))
radius = int(radius)
cv2.circle(img_circle, center, radius, (0, 255, 0), 2)
cv2.circle(img_circle,center,5,(0,255,255),3)
cv2.line(img_circle, center, (center[0], center[1]+radius), (0,0,255), 2)
cv2.line(img_circle, center, (center[0], center[1]-radius), (0,0,255), 2)
cv2.line(img_circle, center, (center[0]+radius, center[1]), (0,0,255), 2)
cv2.line(img_circle, center, (center[0]-radius, center[1]), (0,0,255), 2)
cv2.putText(img_circle, "(X:"+str(l)+","+"Y:"+str(k)+")" ,(x,y-10),cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,255),2)
cv2.putText(img_circle, "Radius:"+str(radius) ,(x,y-60),cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,255),2)
```

9-) Aspect Ratio(En-Boy oranı)

- Görüntü işlemede, bir nesnenin "aspect ratio" olarak adlandırılan en-boy oranı, nesnenin genişliği ile yüksekliği arasındaki oranı ifade eder.
- Aspect ratio, nesnenin şeklini ve görüntüdeki proporsiyonlarını belirlemek için kullanılan bir ölçüdür.
- Aspect ratio, bir nesnenin genellikle yatay veya dikey uzunluğunu ifade eden değeri temsil eder.
- Örneğin, bir dikdörtgen şeklindeki bir nesne için aspect ratio, genişlik ve yükseklik arasındaki oranı ifade eder.
- Eğer aspect ratio değeri 1'den farklı ise, nesne dikdörtgen şeklinde kabul edilir.
- Aspect ratio değeri 1'e yaklaştıkça nesne kareye yakınlaşır, değer 1'e ne kadar yakınsa o kadar simetrik kabul edilir.

- Aspect ratio, nesne tanıma, nesne izleme, nesne sınıflandırma ve görüntü hizalama gibi birçok görüntü işleme uygulamasında kullanılır.
- Özellikle nesne tanıma ve sınıflandırmada, aspect ratio değeri nesnelerin şekil özelliklerini belirlemek ve farklı sınıfları ayırt etmek için kullanılan bir parametre olabilir.



$$\text{Aspect Ratio} = \frac{\text{Bounding Rectangle Genişliği (Width)}}{\text{Bounding Rectangle Yüksekliği (Height)}}$$

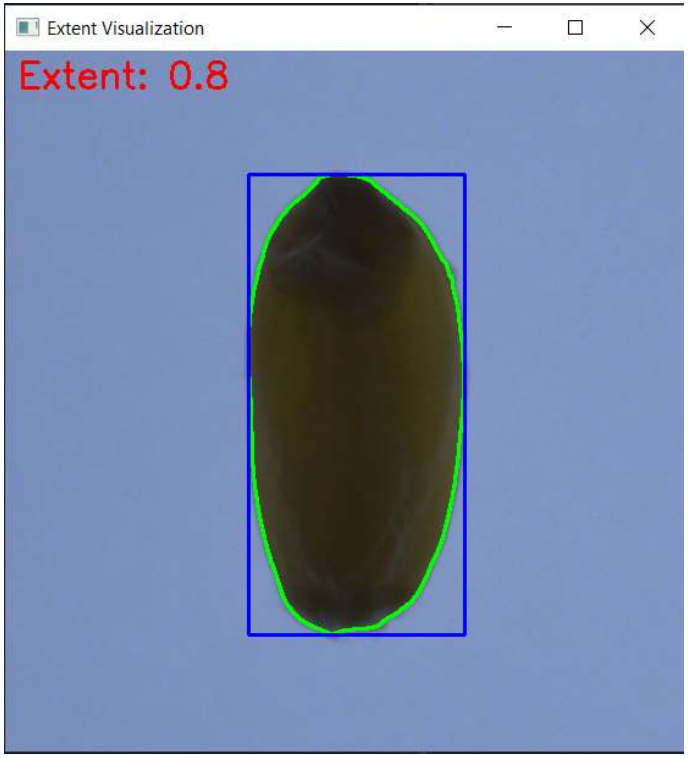
```
# Aspect Ratio
x,y,w,h = cv2.boundingRect(contour)
aspect_ratio = float(w)/h
print(aspect_ratio)
```

```
0.46949602122015915
```

10-) Extent (Kapsam)

- Extent, bir nesnenin yüzde cinsinden ifade edilen alanını ifade eder.
- Nesnenin tüm piksellerinin yer aldığı alanın, toplam görüntü alanına oranı olarak hesaplanır.
- Extent değeri 0 ile 1 arasında değişir, 1 nesnenin tamamını temsil ederken, 0 nesnenin hiçbir pikselinin olmadığı anlamına gelir.
- Extent, nesne segmentasyonu, nesne tanıma ve sınıflandırma gibi görüntü işleme uygulamalarında kullanılır.
- Nesne segmentasyonunda, nesnelerin piksellerini belirlemek için kullanılan bir ölçüdür.

- Nesne tanıma ve sınıflandırmada ise, nesnenin görüntü içerisindeki varlığını ve boyutunu belirlemek için kullanılır.
- Extent değeri, nesnenin büyüklüğü ve yoğunluğu hakkında bilgi sağlar ve nesneleri diğerlerinden ayırt etmek ve analiz etmek için kullanılan bir özelliktir.



$$\text{Extent} = \frac{\text{Kontur Alanı (Area of Contour)}}{\text{Bounding Rectangle Alanı (Area of Bounding Rectangle)}}$$

```
 #Extent
area = cv2.contourArea(contour)
x,y,w,h = cv2.boundingRect(contour)
rect_area = w*h
extent = float(area)/rect_area
print(extent)
```

```
0.8008736831062956
```

11-) Solidity

- Görüntü işlemede, "solidity" terimi, bir nesnenin alanının, onu çevreleyen sınıra (boundary) oranını ifade eder.
- Solidity, nesnenin dolgunluğunu veya katılığını temsil eder.
- Solidity değeri, bir nesnenin alanını, onu çevreleyen sınıra bölerek elde edilir.
- Yani, nesnenin tüm piksellerinin kapladığı alanın, sınırlayıcı konturunun kapladığı alanla oranını ifade eder.
- Solidity değeri 0 ile 1 arasında değişir, 1 nesnenin tamamen sınıra oturduğunu gösterirken, 0 ise nesnenin içinde hiçbir pikselin sınıra dokunmadığını ifade eder.

- Solidity, nesnenin yoğunluğunu ve şeklini ifade eder.
- Yüksek bir solidity değeri, nesnenin daha dolgun ve sınıra daha yakın olduğunu gösterirken, düşük bir değer daha boşluklu veya delikli bir yapıya işaret eder.
- Nesne tanıma ve sınıflandırmada, nesnenin şekil özelliklerini belirlemek ve nesne sınıflarını ayırt etmek için kullanılan bir ölçüdür.
- Nesne segmentasyonunda ise, nesnelerin sınırlarını belirlemek ve arka plana göre ayrım yapmak için kullanılır.

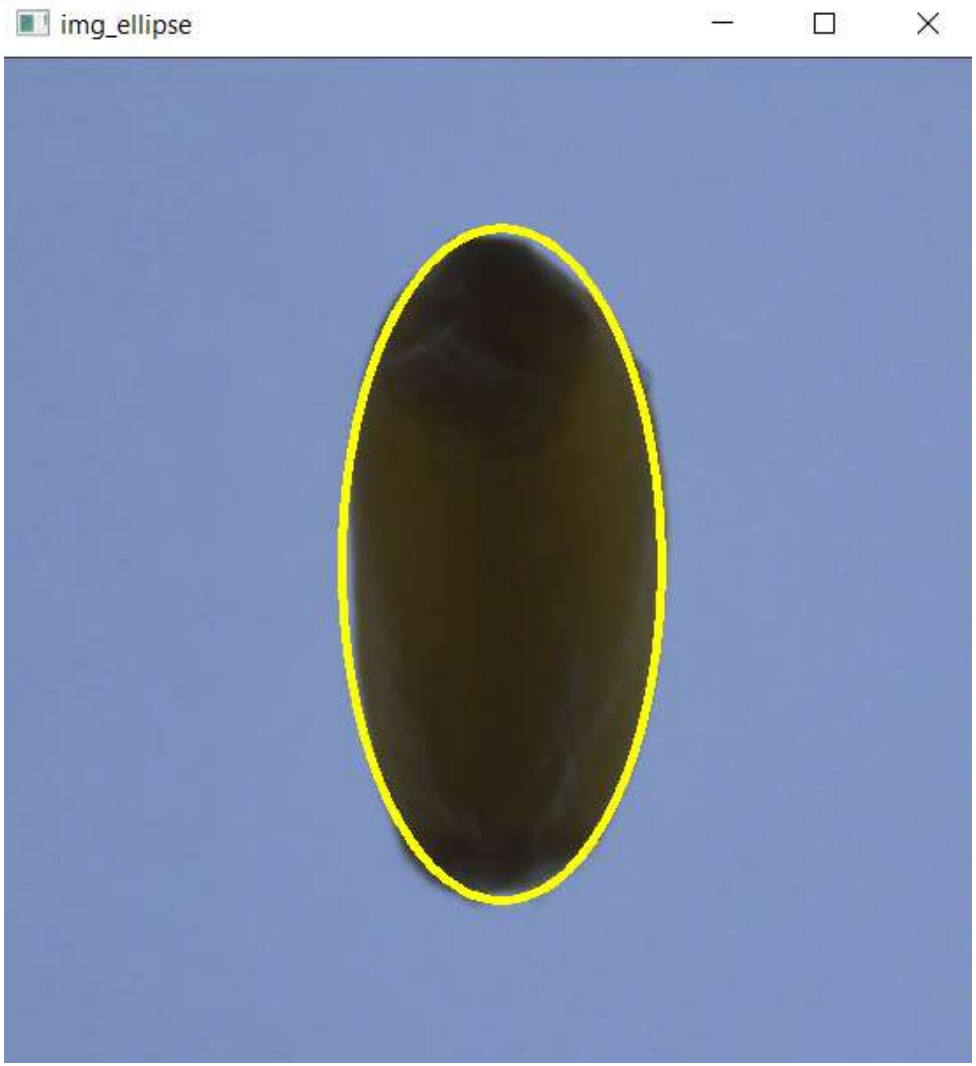
$$\text{Solidity} = \frac{\text{Kontur Alanı (Area of Contour)}}{\text{Konveks Gövde Alanı (Area of Convex Hull)}}$$


```
#Solidity
area = cv2.contourArea(contour)
hull = cv2.convexHull(contour)
hull_area = cv2.contourArea(hull)
solidity = float(area)/hull_area
print(solidity)
```

```
0.9930503293660748
```

12-) Ellipse

- "Ellipse" terimi, bir nesnenin en iyi uyan elips şeklini ifade eder.
- Nesnenin genellikle karmaşık şeklini daha basit bir geometrik şekille temsil etmek amacıyla kullanılır.
- Ellipse, bir nesnenin en uyumlu elips şeklini ifade ederken, bu elips, nesnenin alanını ve şeklini en iyi şekilde yansıtmaya çalışır.
- Nesne üzerindeki piksellerin dağılımı ve konumu dikkate alınarak, uygun bir elips parametresi belirlenir.
- Bu parametreler genellikle elipsin merkezi, yarı büyük eksen uzunluğu, yarı küçük eksen uzunluğu ve eksenlerin yönelimi gibi değerleri içerir.

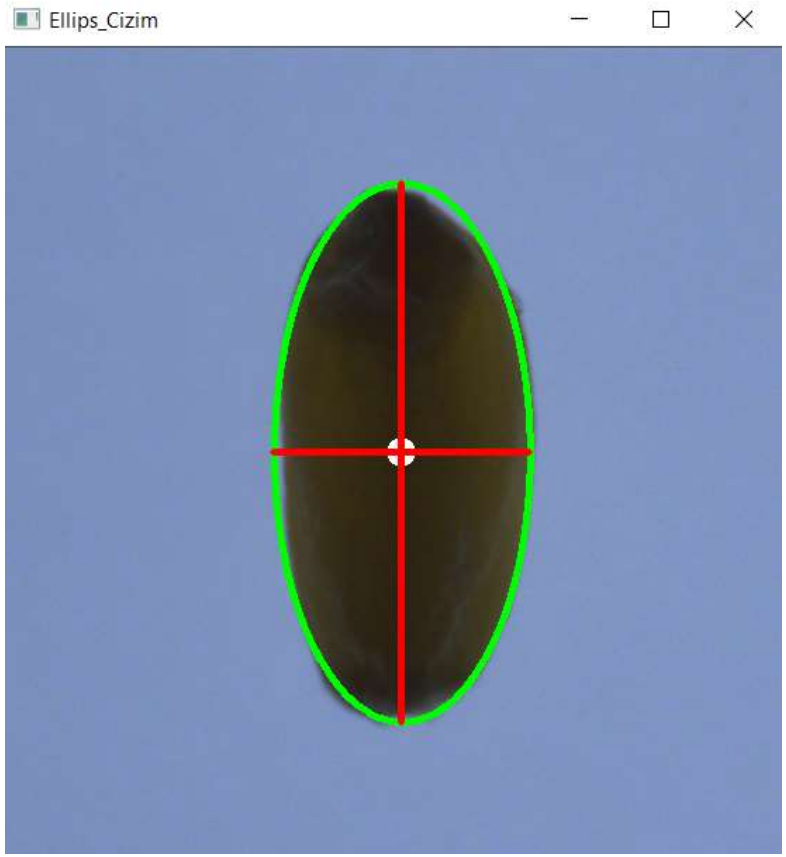



```
img_ellipse=img.copy()
ellipse = cv2.fitEllipse(contour)
cv2.ellipse(img_ellipse,ellipse,(0,255,255),3)
```

```
#Orientation
(x,y),(MA,ma),angle = cv2.fitEllipse(contour)
```

- cv2.fitEllipse(contour) işlevi, kontur verisini alır ve bu kontura en iyi uyan elipsin parametrelerini döndürür.
- Bu parametreler genellikle bir elipsin merkezi, yarı büyük eksen uzunluğu, yarı küçük eksen uzunluğu ve eksenlerin yönelimini içerir.
- Döndürülen değer, genellikle ((center_x, center_y), (major_axis_length, minor_axis_length), angle) formatında bir tuple'dır.
- Bu değerlerin anlamı şu şekildedir:
- (center_x, center_y): Elipsin merkezinin koordinatları.
- (major_axis_length, minor_axis_length): Yarı büyük ekseni ve yarı küçük ekseni temsil eden uzunluk değerleri.
- angle: Elipsin eksenlerinin yönelimini ifade eden açı değeri.

```python
ellipse = cv2.fitEllipse(contour)
(xc,yc),(d1,d2),angle = ellipse
#print(xc,yc,d1,d1,angle)

# draw ellipse
result = img.copy()
cv2.ellipse(result, ellipse, (0, 255, 0), 3)
# draw circle at center
xc, yc = ellipse[0]
cv2.circle(result, (int(xc),int(yc)), 10, (255, 255, 255), -1)

# draw vertical line
# compute major radius
rmajor = max(d1,d2)/2
if angle > 90:
    angle = angle - 90
else:
    angle = angle + 90
#print(angle)
xtop = xc + math.cos(math.radians(angle))*rmajor
ytop = yc + math.sin(math.radians(angle))*rmajor
xbot = xc + math.cos(math.radians(angle+180))*rmajor
ybot = yc + math.sin(math.radians(angle+180))*rmajor
cv2.line(result, (int(xtop),int(ytop)), (int(xbot),int(ybot)), (0, 0, 255), 3)


rmajor = min(d1,d2)/2
if angle > 90:
    angle = angle - 90
else:
    angle = angle + 90
#print(angle)
xtop = xc + math.cos(math.radians(angle))*rmajor
ytop = yc + math.sin(math.radians(angle))*rmajor
xbot = xc + math.cos(math.radians(angle+180))*rmajor
ybot = yc + math.sin(math.radians(angle+180))*rmajor
cv2.line(result, (int(xtop),int(ytop)), (int(xbot),int(ybot)), (0, 0, 255), 3)
```



13-) Eccentricity

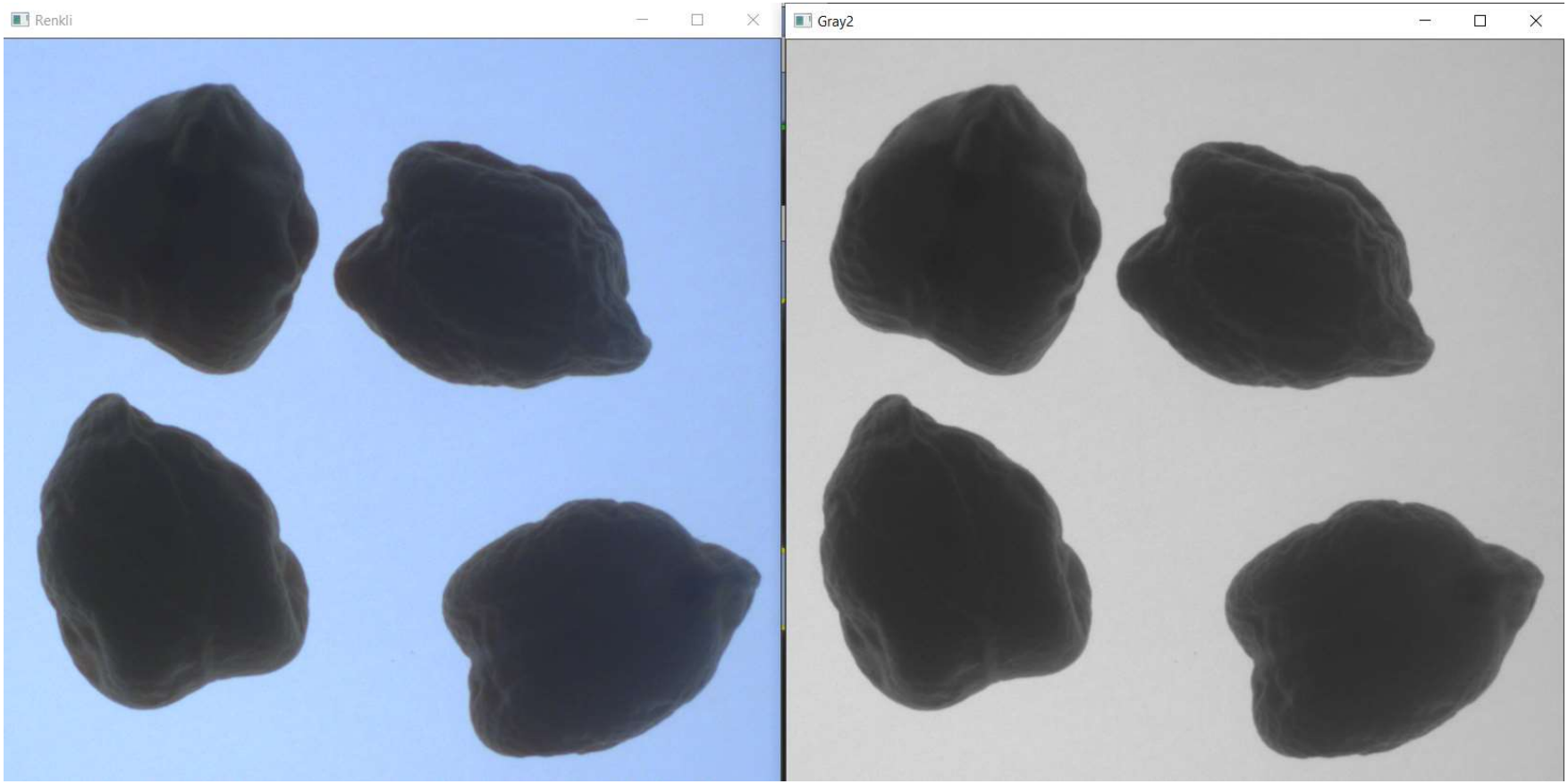
- Eccentricity, eksantriklik, dışmerkezlilik, dairesellikten uzaklık
- "eccentricity" terimi, bir nesnenin elips şeklinin ne kadar uzandığını veya yuvarlaklıktan ne kadar uzaklaştığını ifade eder.
- Eccentricity, bir elipsin yükseklik ve genişlik oranına dayanan bir ölçüdür.
- Eccentricity değeri 0 ile 1 arasında değişir. 0, bir daireyi temsil ederken, 1 ise en uzun ekseni ile en kısa ekseni arasındaki oranı ifade eder.
- Dolayısıyla, eccentricity değeri ne kadar yaklaşık 1 ise, elips o kadar uzamış veya yuvarlaktan uzaklaşmış demektir.

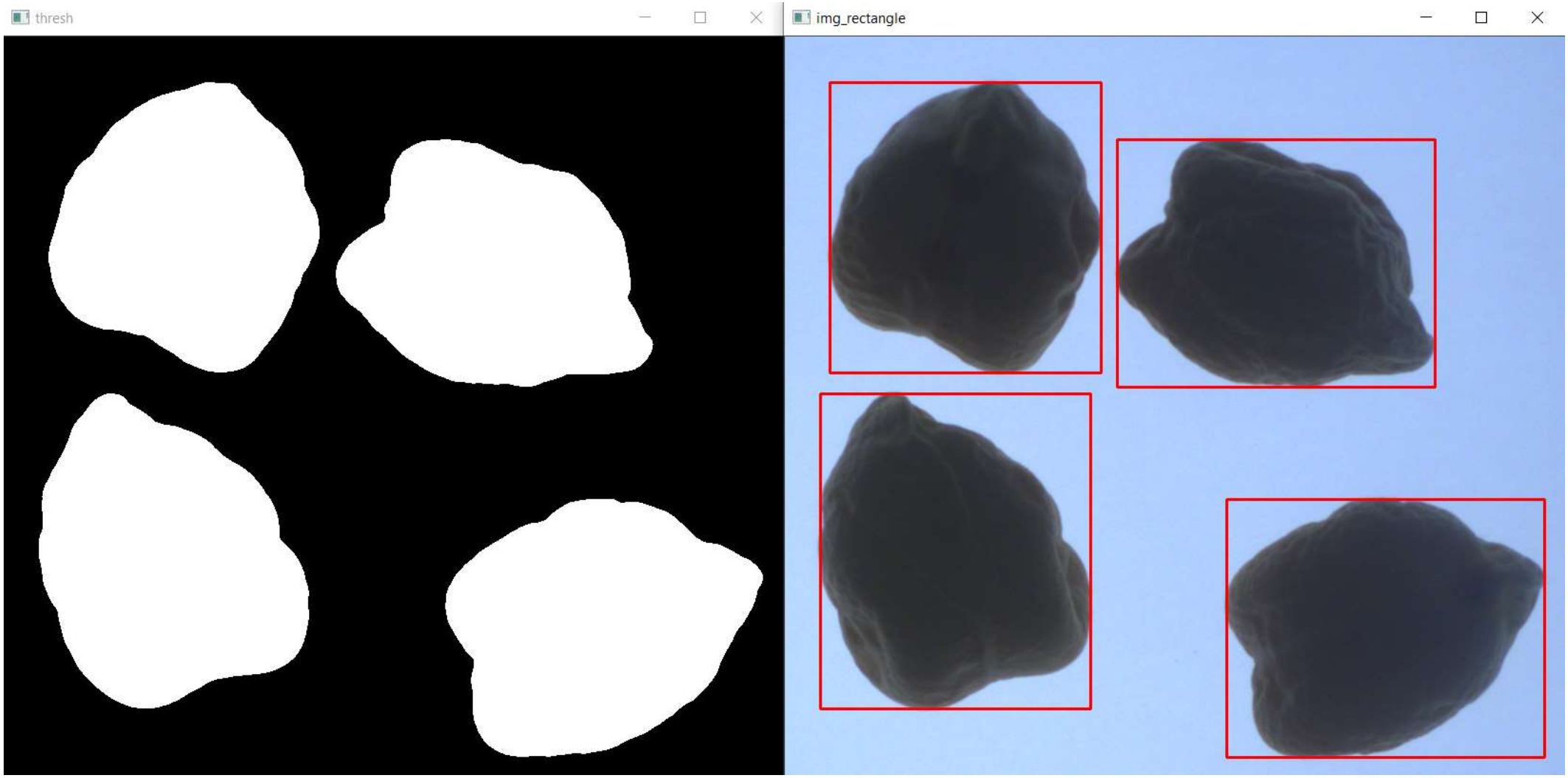
•**Eccentricity düşük** (≈0): Şekil yuvarlak bir daireye yakındır.
•**Eccentricity yüksek** (≈1): Şekil çok ince, uzun ve çizgiye yakındır.

```
#Eccentricity
(x,y),(MA,ma),angle = cv2.fitEllipse(contour)

a = ma/2
b = MA/2

eccentricity = math.sqrt(pow(a,2)-pow(b,2))
eccentricity = round(eccentricity/a,2)
```

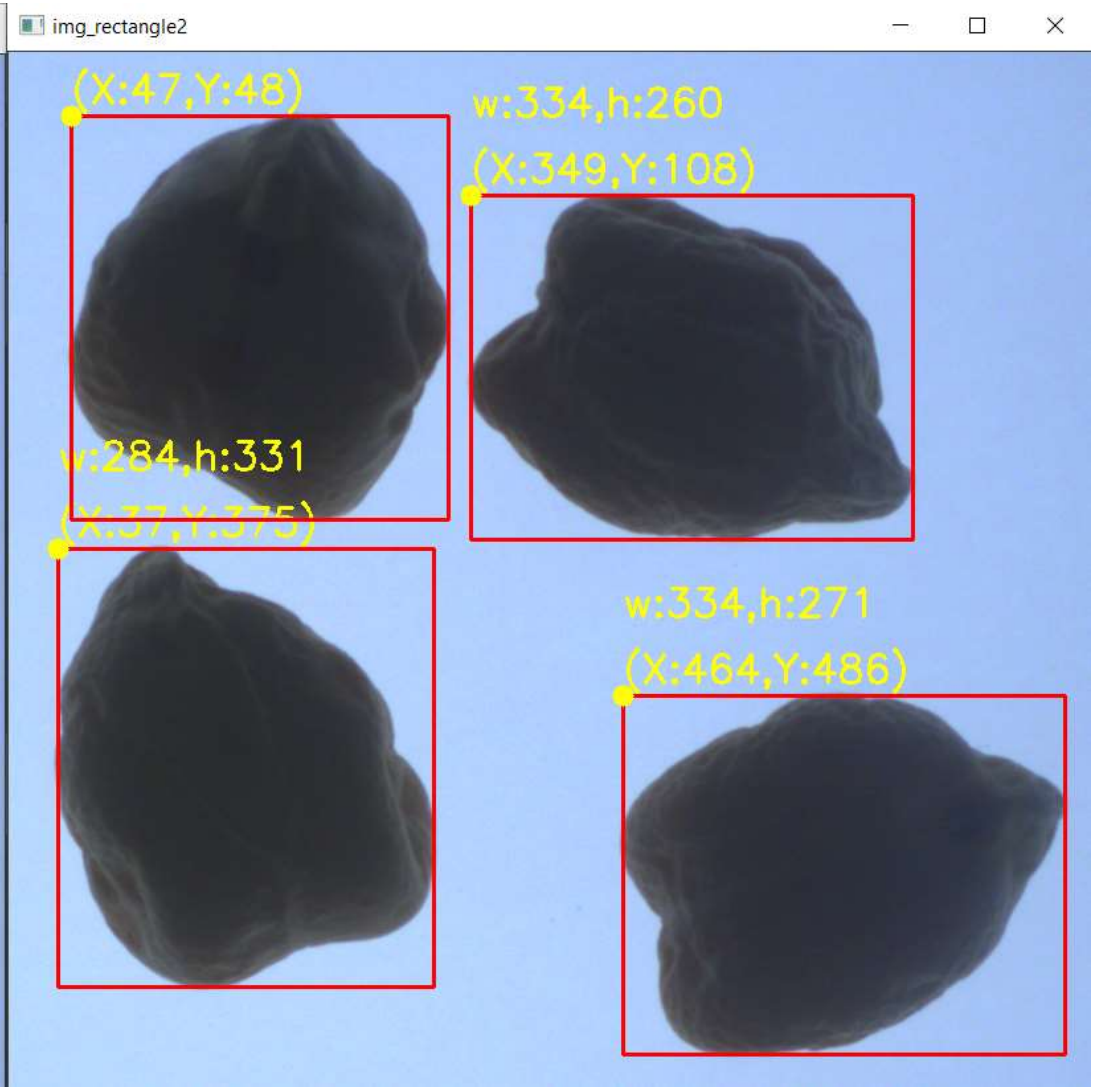
```
0.88
```

14-) Çoklu Nesne Özellikleri

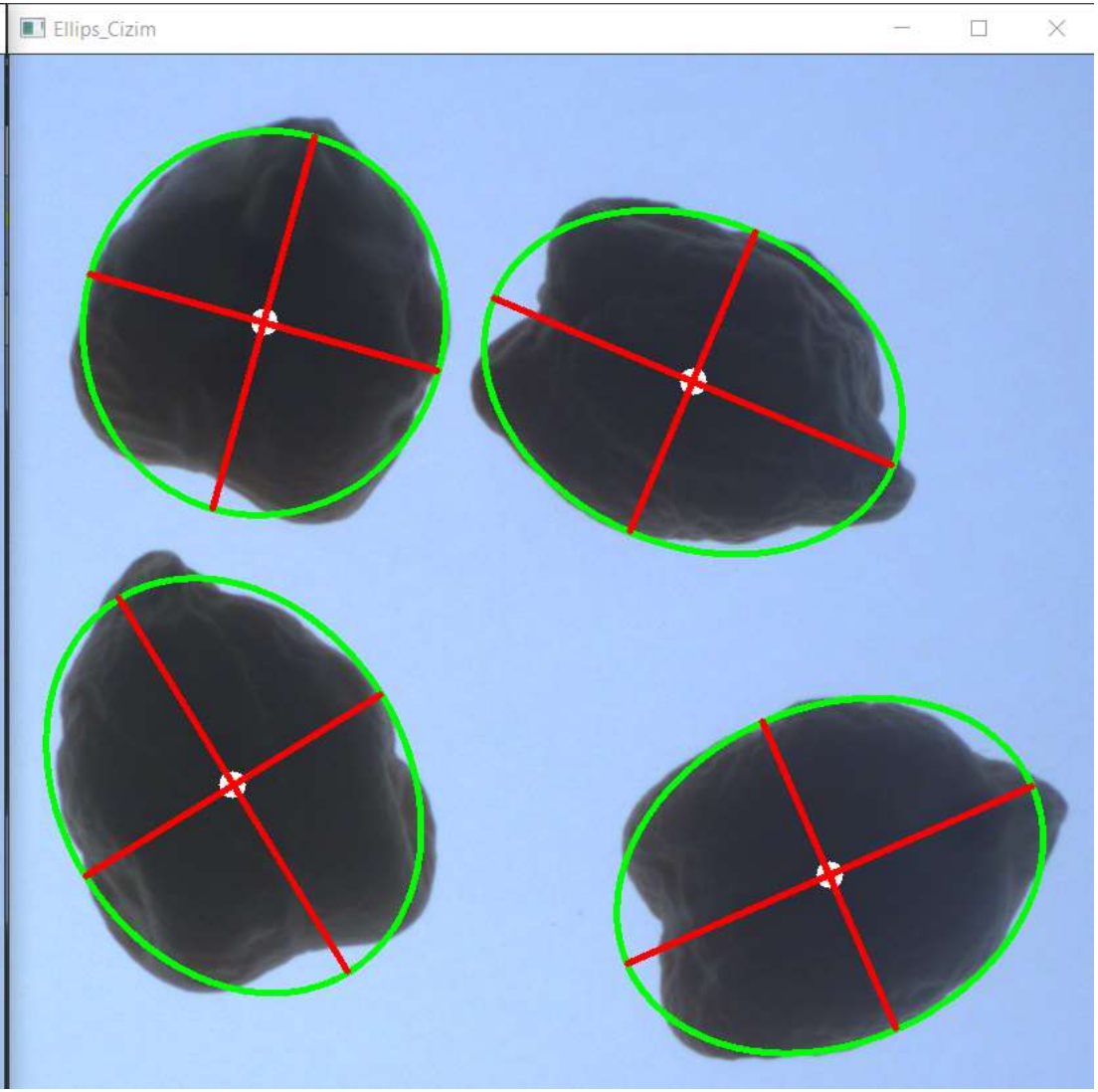


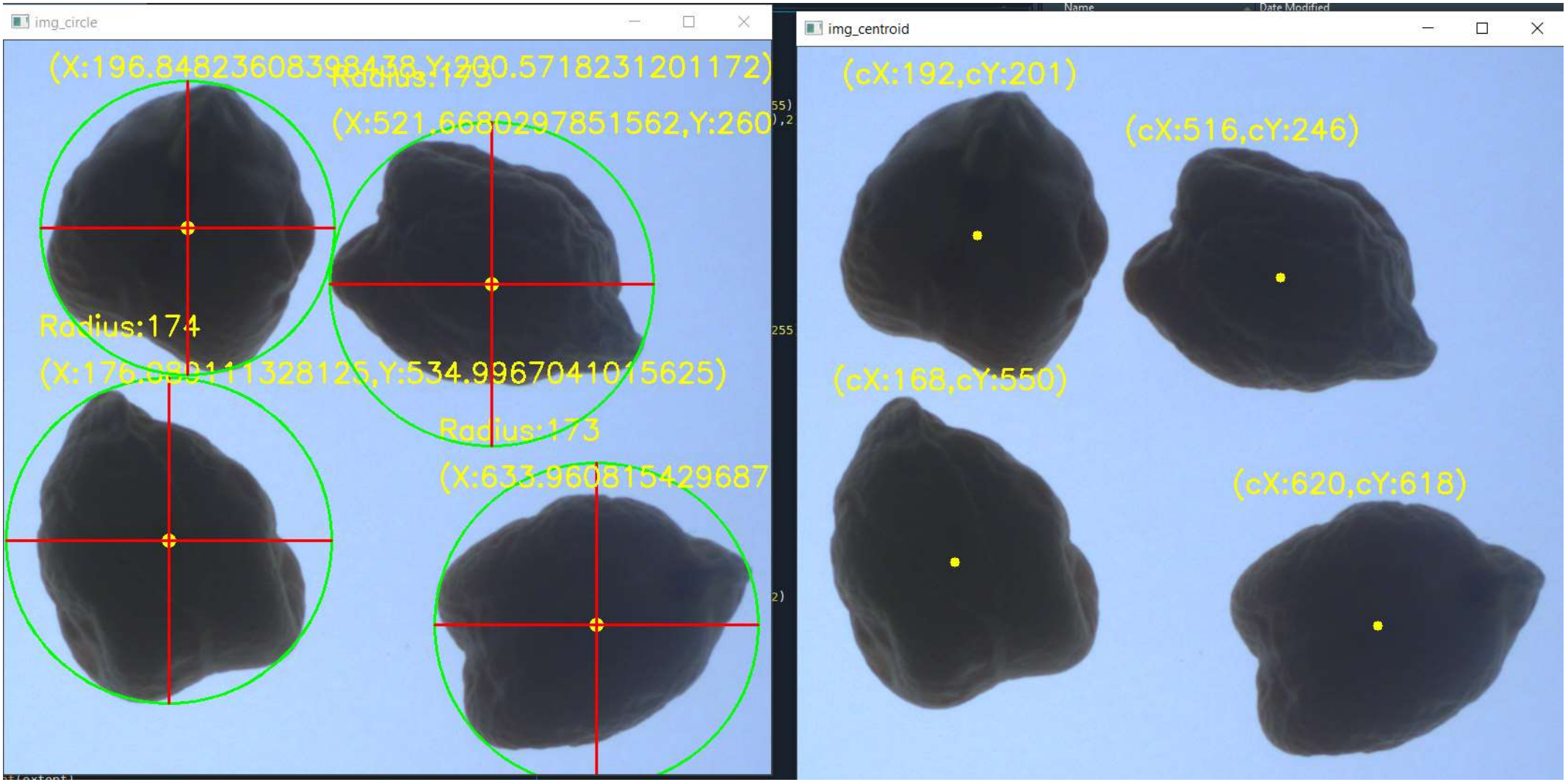

thresh
img_rectangle

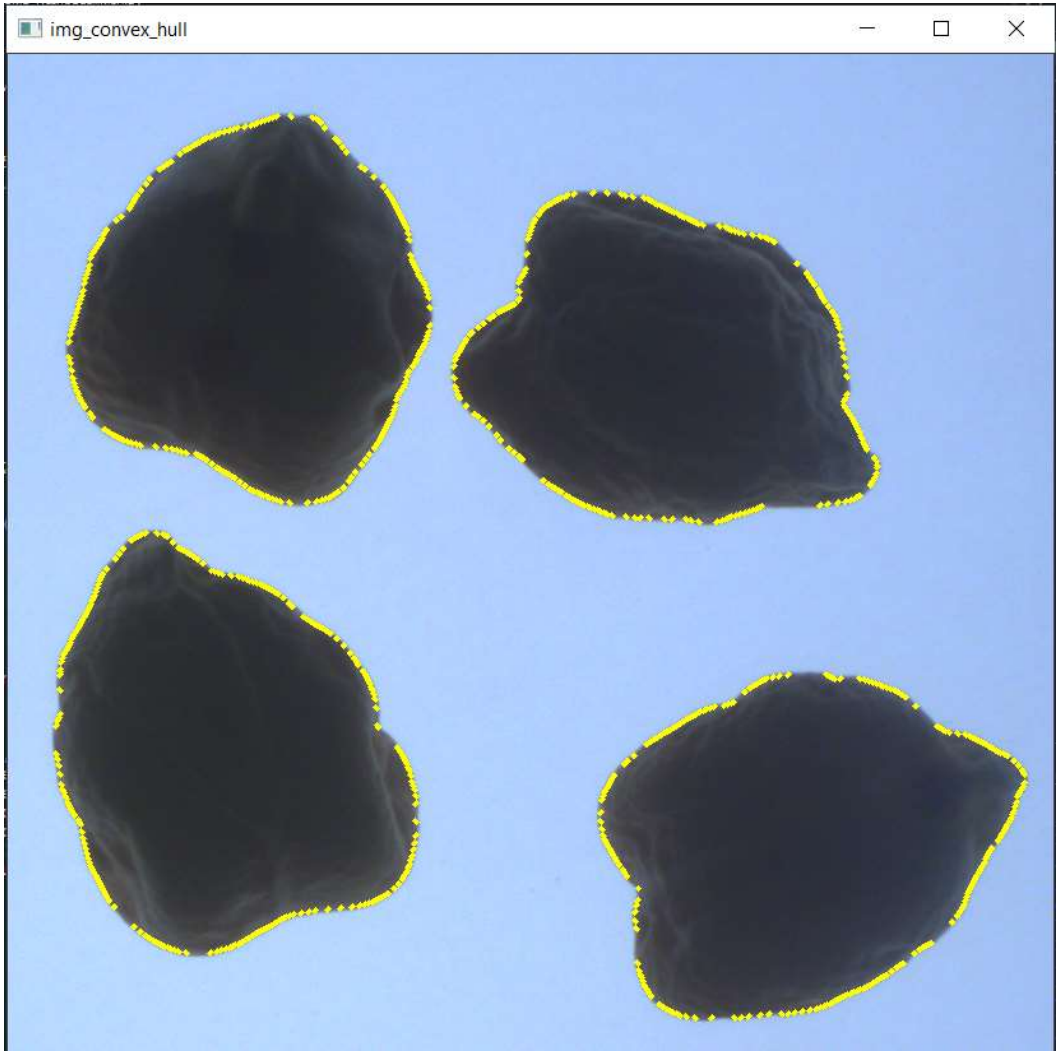

img_cevre


img_rectangle2
(X:47,Y:48)
w:334,h:260
(X:349,Y:108)
w:284,h:331
(X:37,Y:375)
w:334,h:271
(X:464,Y:486)


img_ellipse


Ellips_Cizim


img_circle
(X:196.84823608398438,Y:200.57182312011719)
Radius:173
(X:521.6680297851562,Y:260
Radius:174
(X:176.
1328125,Y:534.9967041015625)
Radius:173
(X:633.960815429687
img_centroid
(cX:192,cY:201)
(cX:516,cY:246)
(cX:168,cY:550)
(cX:620,cY:618)


img_convex_hull


img_Nesne_Sayisi
4
3
2
1

```
NESNE ID: 1
Perimeter: 1003.6021555662155
Alan: 64107.0
Aspect Ratio: 1.2324723247232472
Extent: 0.7082550765627417
Solidity: 0.9752784032130469
Equivalent Diameter: 285.69838552980497
Eccentricity: 0.65
```

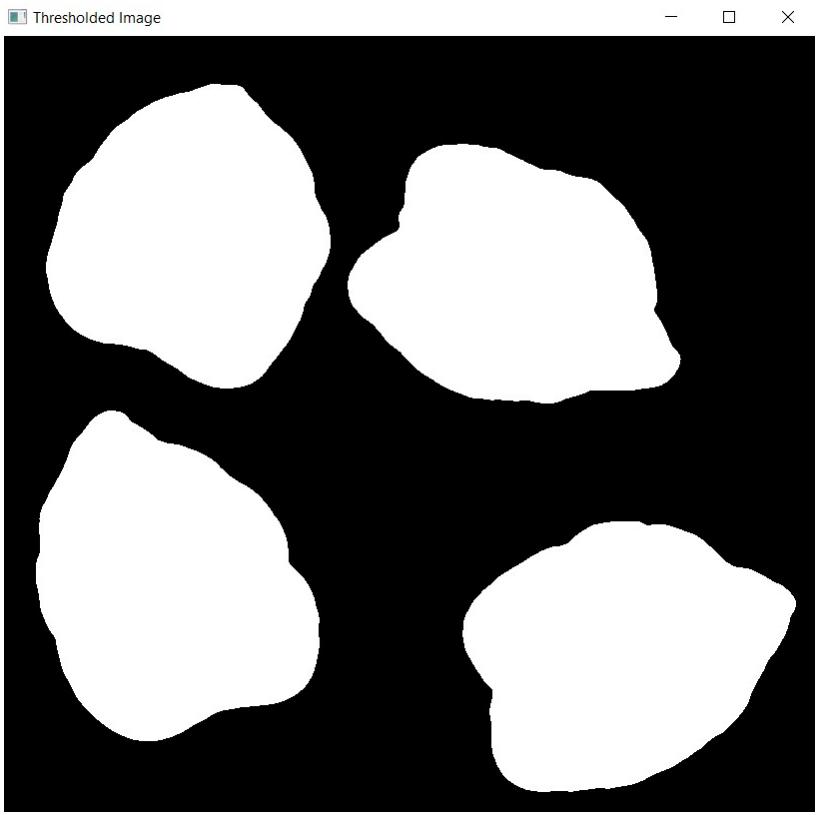
```
NESNE ID: 2
Perimeter: 1018.7737275362015
Alan: 66101.0
Aspect Ratio: 0.8580060422960725
Extent: 0.7031722054380665
Solidity: 0.975696520166795 9
Equivalent Diameter: 290.107578574809
Eccentricity: 0.61
```

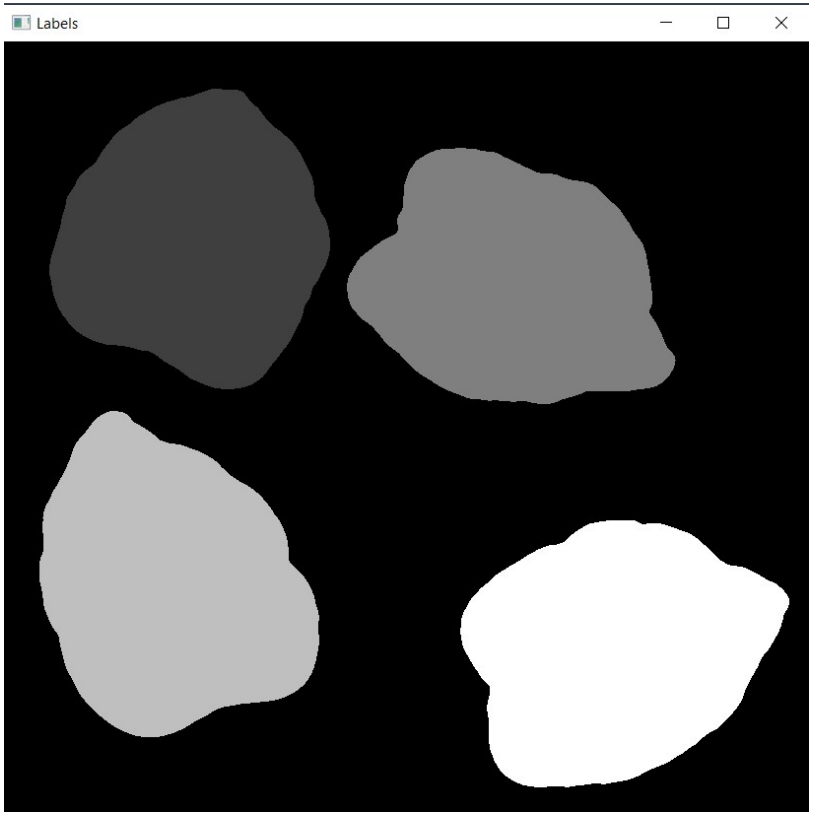
```
NESNE ID: 3
Perimeter: 1000.3473192453384
Alan: 60998.0
Aspect Ratio: 1.284615384615384 7
Extent: 0.7024182404421926
Solidity: 0.9610373241322809
Equivalent Diameter: 278.6845272880349
Eccentricity: 0.66
```

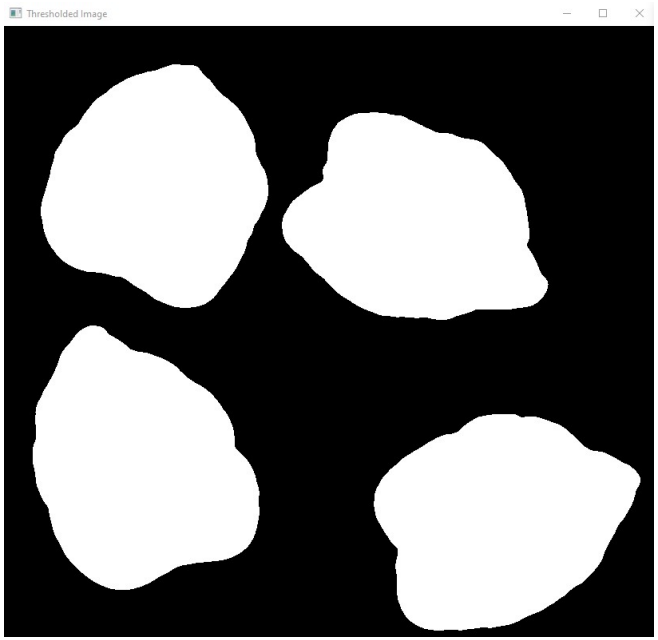
```
NESNE ID: 4
Perimeter: 962.7737267017365
Alan: 61580.0
Aspect Ratio: 0.9344262295081968
Extent: 0.7084268047167098
Solidity: 0.9806357092808459
Equivalent Diameter: 280.0108768687233
Eccentricity: 0.35
```

15-Skimage Yaklaşımı

- Görüntüdeki her bir nesneyi otomatik etiketler ve tüm özellikleri çıkartır





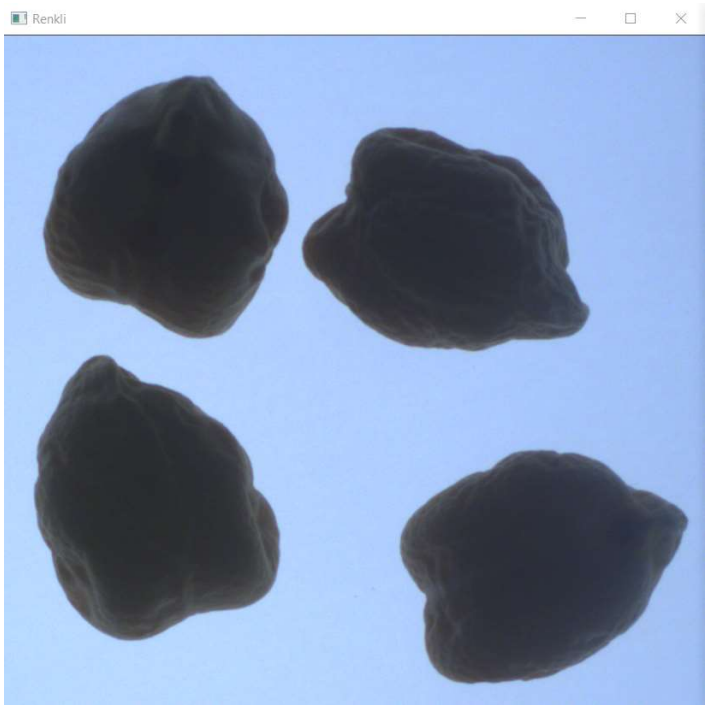


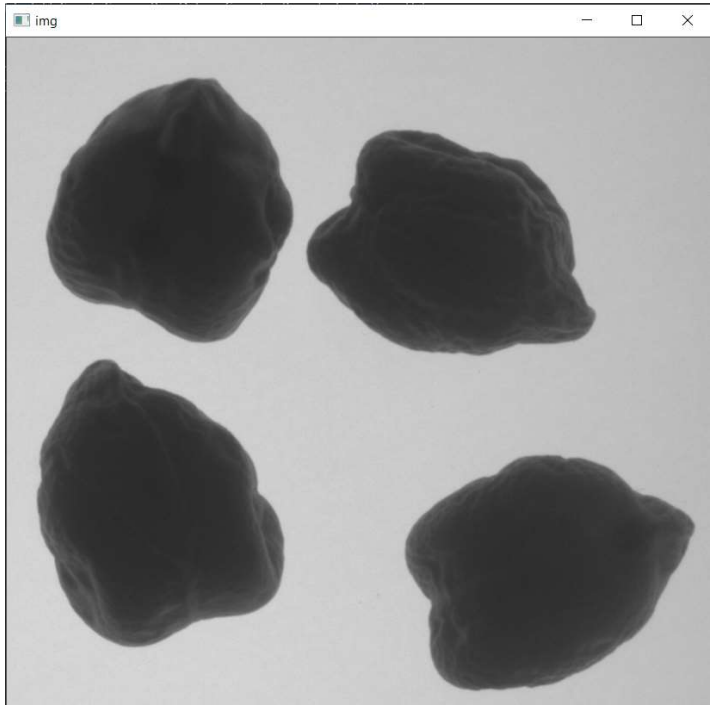
- area
- bbox
- bbox_area
- centroid
- convex_area
- convex_image
- coords
- eccentricity
- equivalent_diameter
- euler_number
- extent
- filled_area
- filled_image
- image
- inertia_tensor
- inertia_tensor_eigvals
- label
- local_centroid
- major_axis_length
- minor_axis_length
- max_intensity
- mean_intensity
- min_intensity
- moments
- moments_central
- moments_hu
- moments_normalized
- orientation
- perimeter
- perimeter_crofton
- solidity
- weighted_centroid
- weighted_local_centroid
- weighted_moments
- weighted_moments_central
- weighted_moments_hu
- weighted_moments_normalized

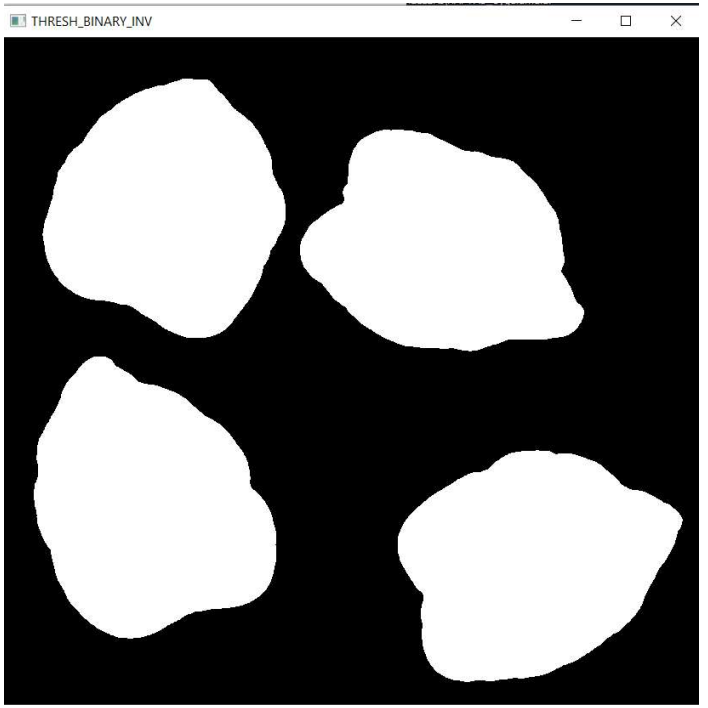
Skimage ile her bir nesneyi etiketle, yani bir ID ata

Regionprops ile her bir nesnenin özelliklerini hesapla, ihtiyaca göre kullan

Her bir nesneye ait 37 özellik hesaplanabiliyor



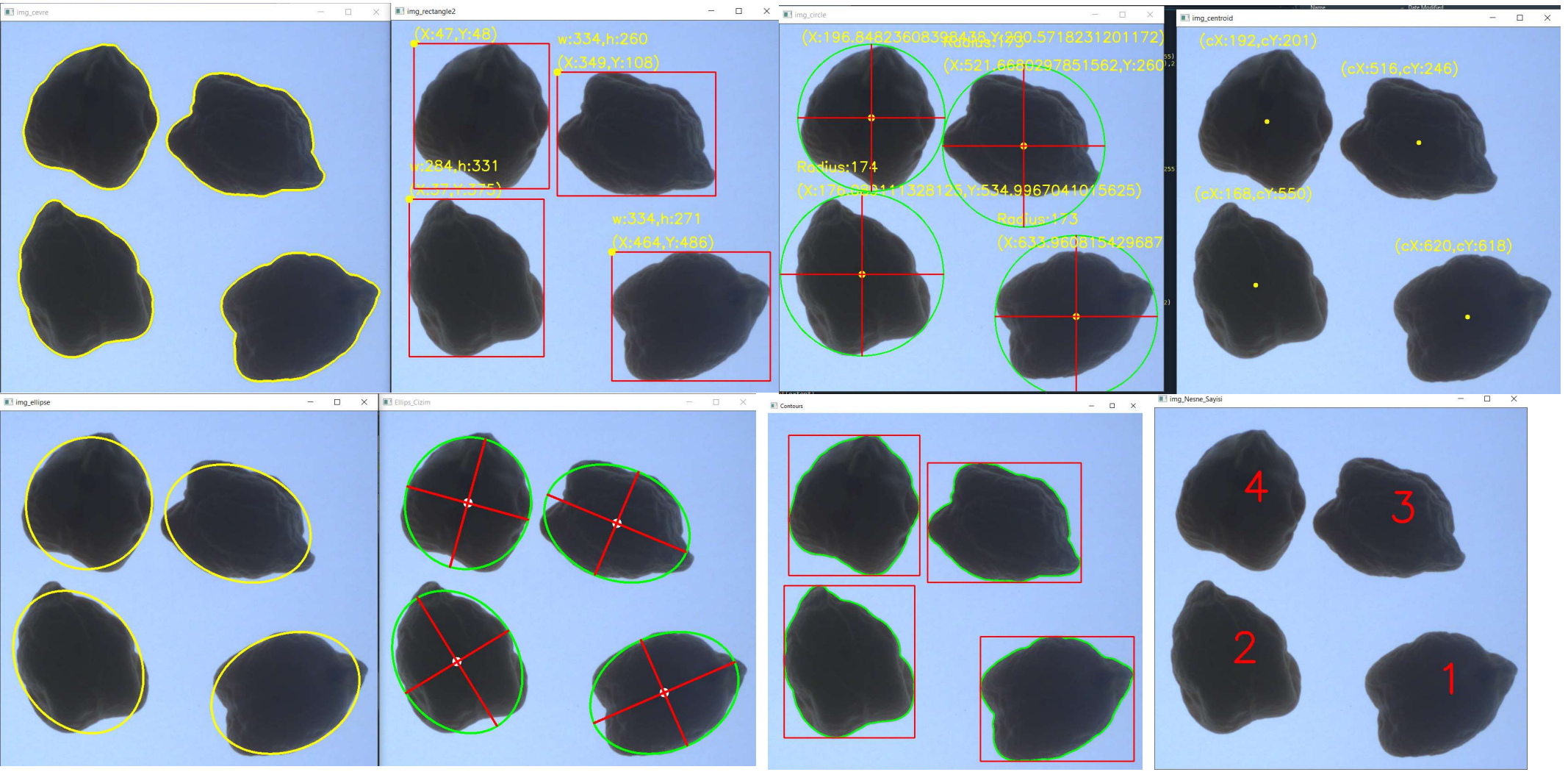




1-Doğru Kamera ve Lens
2-Doğru Aydınlatma
3-Renkli veya Gri Görüntü Al

1-Gri Seviyeye Dönüştür
2-Yumuşatma Filtresi Uygula
(Medyan, Gausssian vb.)

1-Eşikleme Yap
2-HSV uzayı ile arka planı bastır



OpenCV Kontur metodu ile nesnelerin özelliğini hesapla