Makine Görmesi Final Ödevi

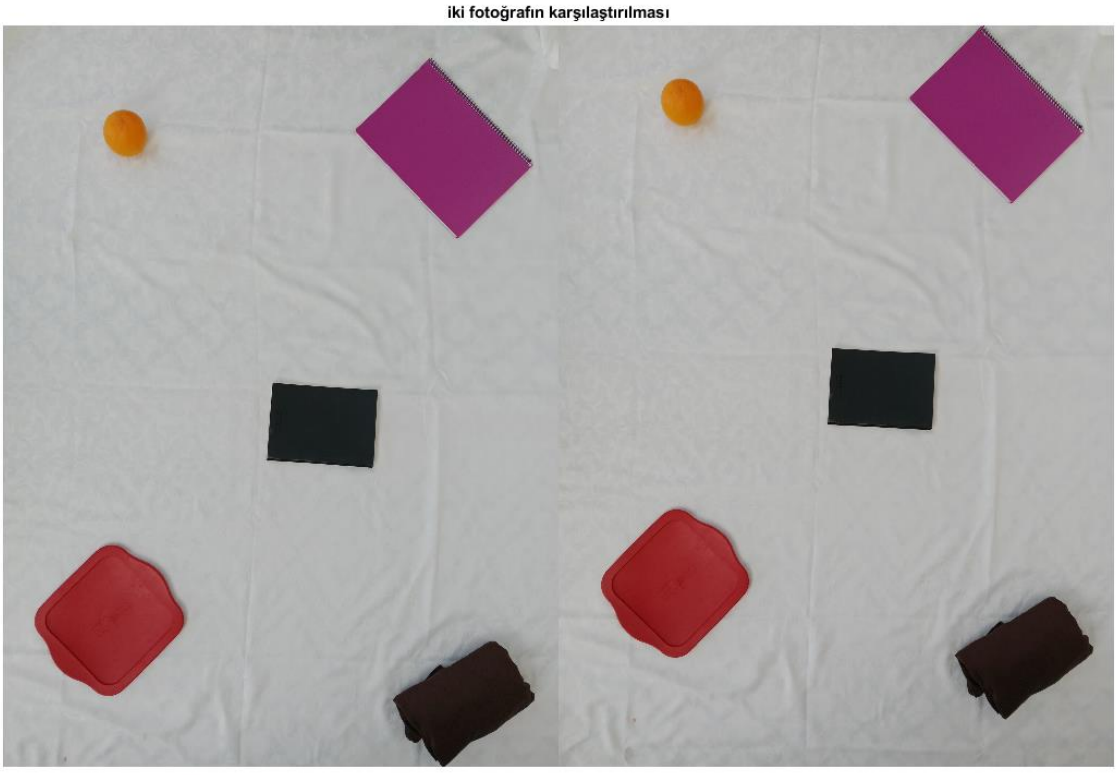
AKDENİZ ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği(YL)

Fahri Kaan Uslu
202451009001

Contents

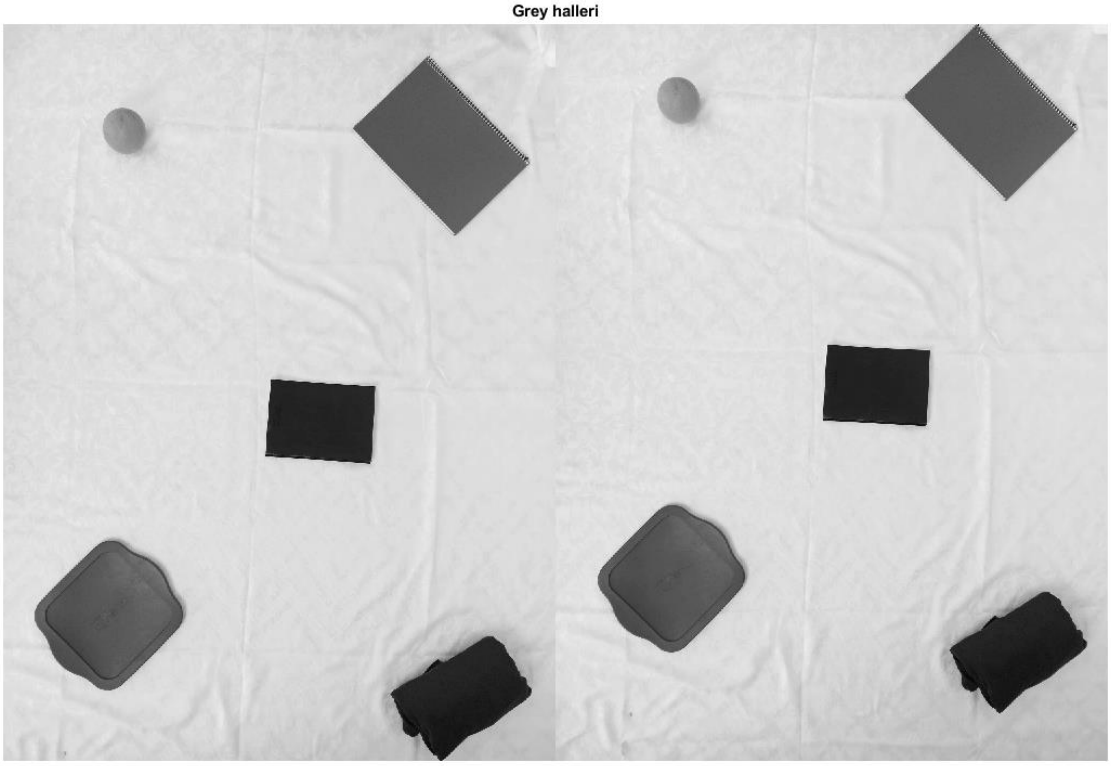
1- Orijinal fotoğrafları okuma & karşılaştırma

```
clc; clear; close all;
mkdir('outputs'); % çıktı klasörü
org_img = iread('or_img.jpeg', 'uint8'); % matrixleri
görmemek için "%" kullandım
shif_img = iread('shif_img.jpeg', 'uint8');
imshowpair(org_img, shif_img,"montage");title("iki
fotoğrafın karşılaştırılması")
s1 = size(org_img) % [4080x3060x3]
s2 = size(shif_img)% [4080,3060,3]
```



2- Histogram & Grey okuma & Adjust etme işlemleri

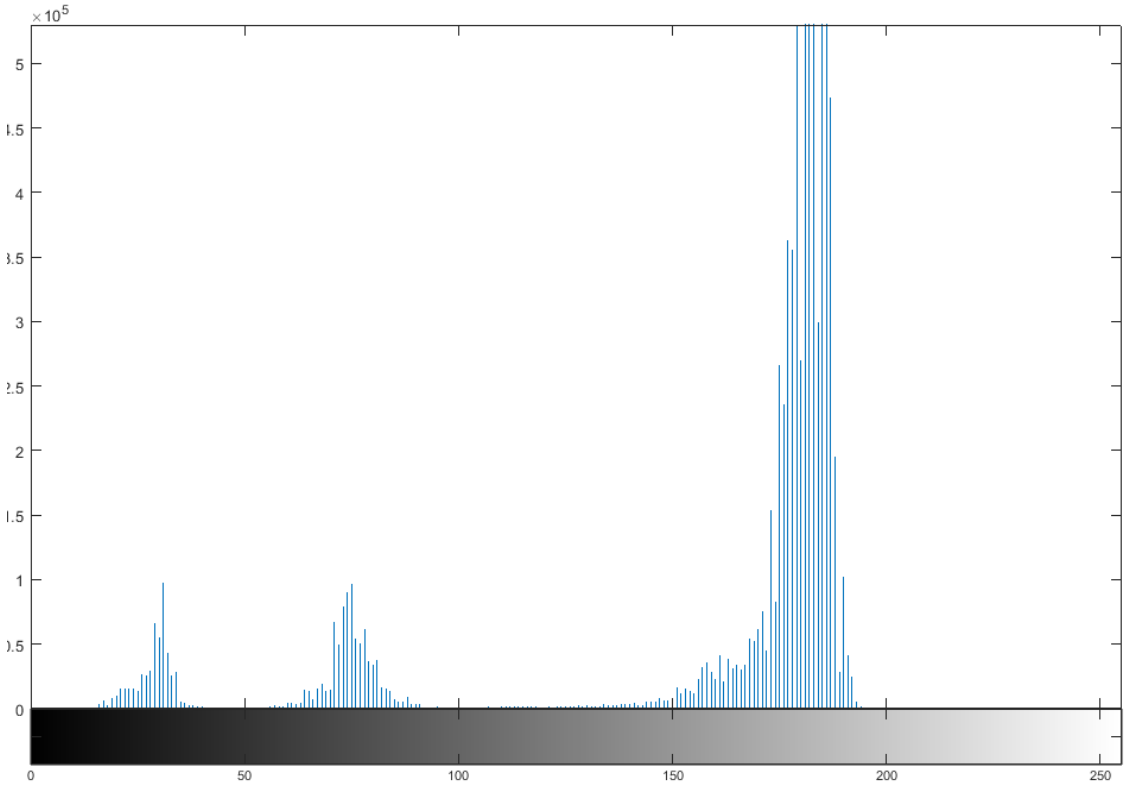
```
org_img_grey = iread('or_img.jpeg', 'uint8', 'grey'); %
matrixleri görmemek için "%" kullandım
shif_img_grey = iread('shif_img.jpeg', 'uint8', 'grey');
imshowpair(org_img_grey, shif_img_grey,
"montage");title("Grey halleri")
```

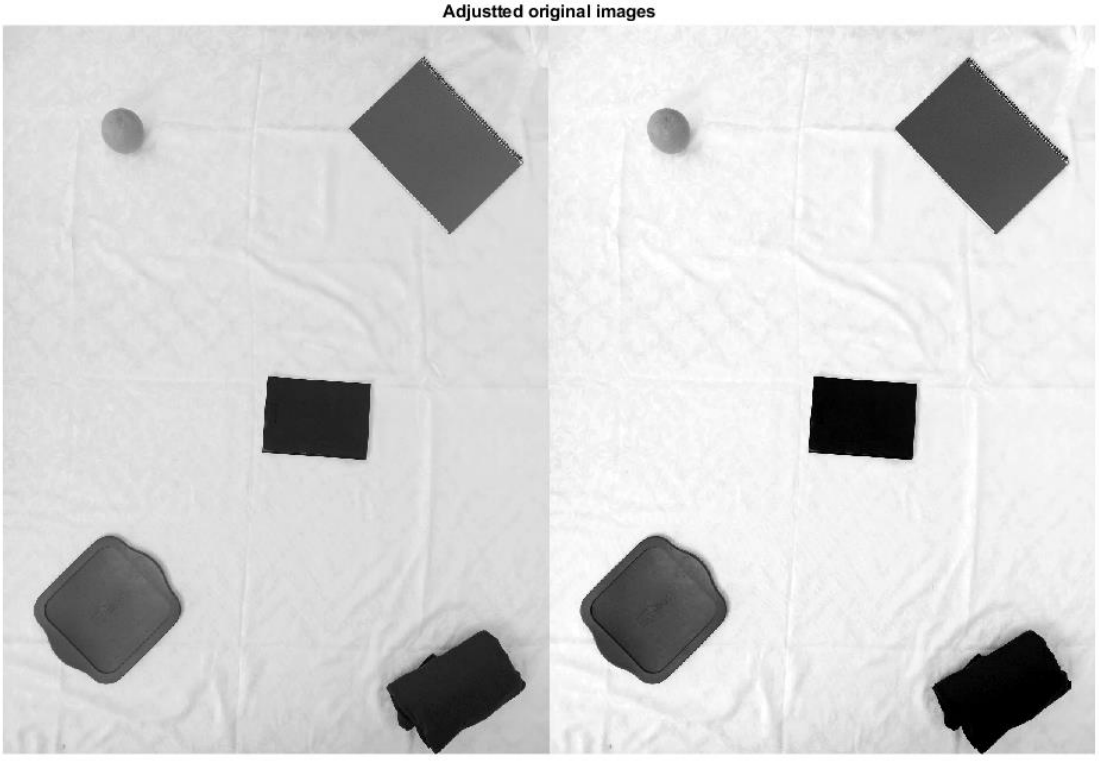


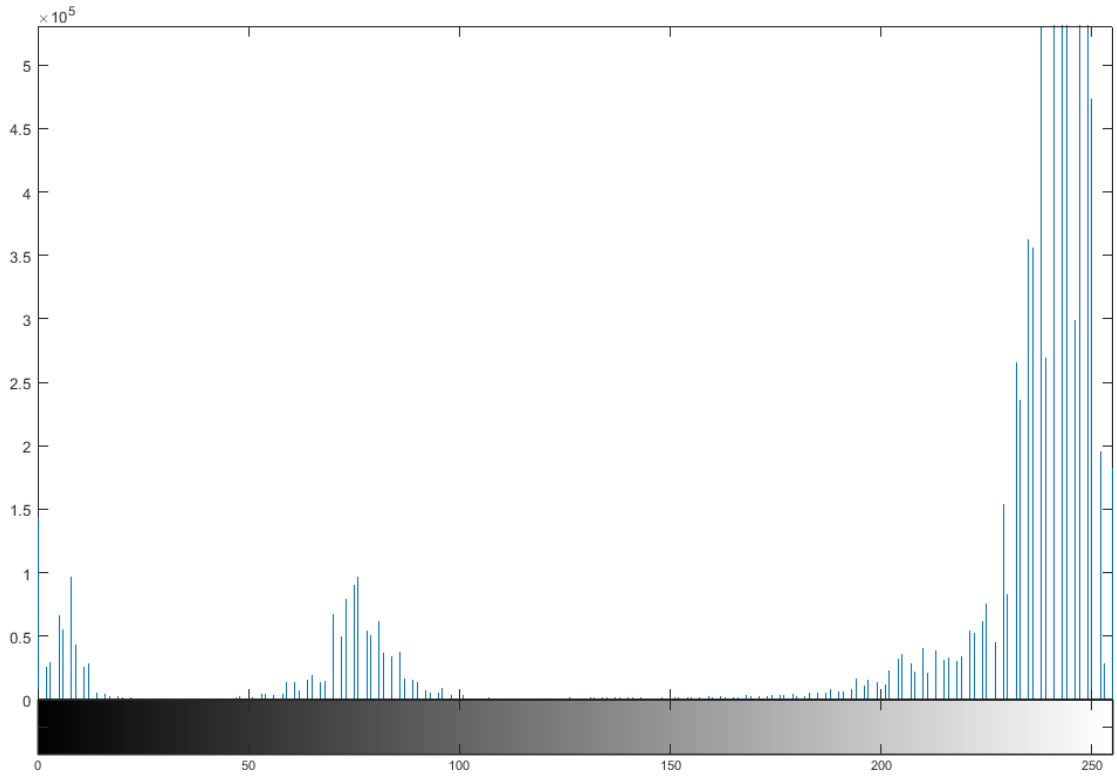
2.1 – Adjusting & histogram analizi

```
% Orijinal image'yi adjust etme ve histogramına bakma
imhist(org_img_grey)
org_img_greyAdj = imadjust(org_img_grey);
imshowpair(org_img_grey, org_img_greyAdj, "montage")
title("Adjustted original images");imhist(org_img_greyAdj)

% Shifted image'yi adjust etme ve histogramına bakma
imhist(shif_img_grey)
shif_img_greyAdj = imadjust(shif_img_grey);
imshowpair(shif_img_grey, shif_img_greyAdj,
"montage");title("Adjustted shifted images")
imhist(shif_img_greyAdj)
```


×10^5
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
50
100
150
200
250

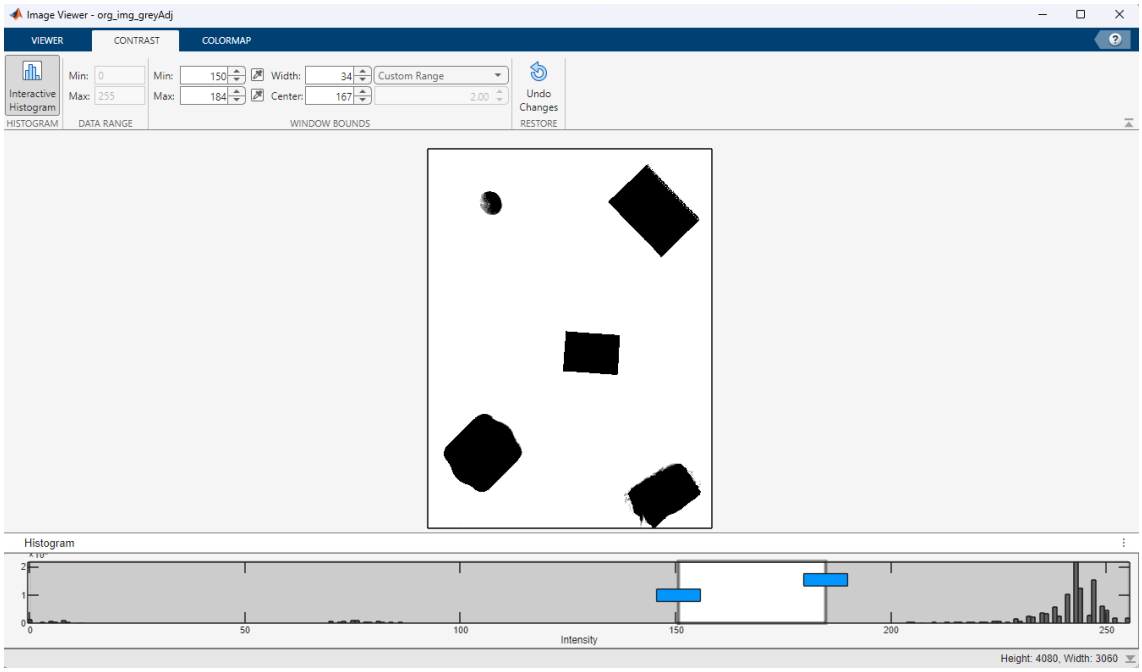

Adjustted original images



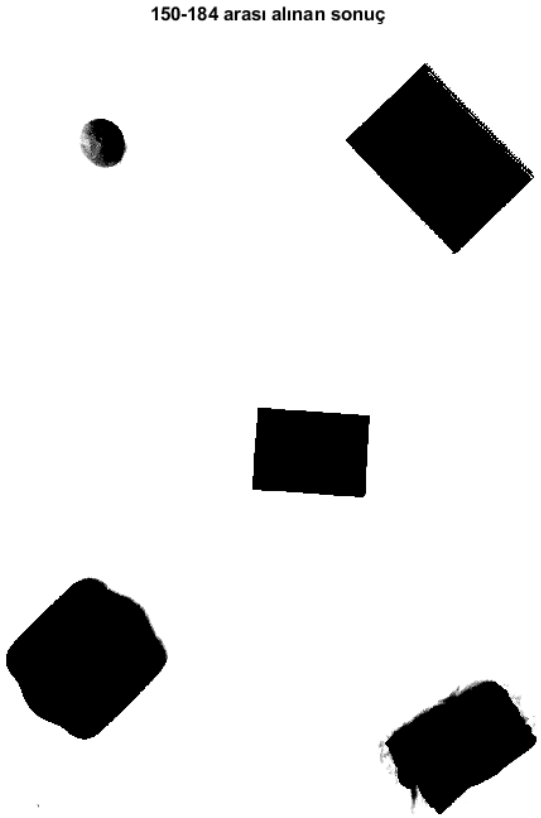
Not : Görüldüğü üzere adjusting yaparak beyazlar daha beyaz siyahlar daha siyah hale geldi. Histogram 'or_img.jpeg'e ait. 'shif_img.jpeg' de aynı olduğu için gösterme gereği duymadım.

3- Histogramı bilinen fotoğrafın belli bir aralığını alma

Not: org_img_greyAdj ve shif_img_greyAdj her ne kadar grey versiyonuna göre çok iyi gözükse de daha iyi hale getirebilmek için histogramları üzerinde oynanabilir. Burada imbinarize() fonksiyonu tercih edilebilir lakin çok iyi bir sonuç çıkarmadığı için aşağıda fotoğrafını vermiş olduğum imageViewer adlı interactive bir matlab app'ini kullandım. Daha sonra imageViewer_or ve imageViewer_shif olarak kaydettim.




```
imshow(imageViewer_or);title("150-184 arası alınan sonuç")
imshow(imageViewer_shif);
```

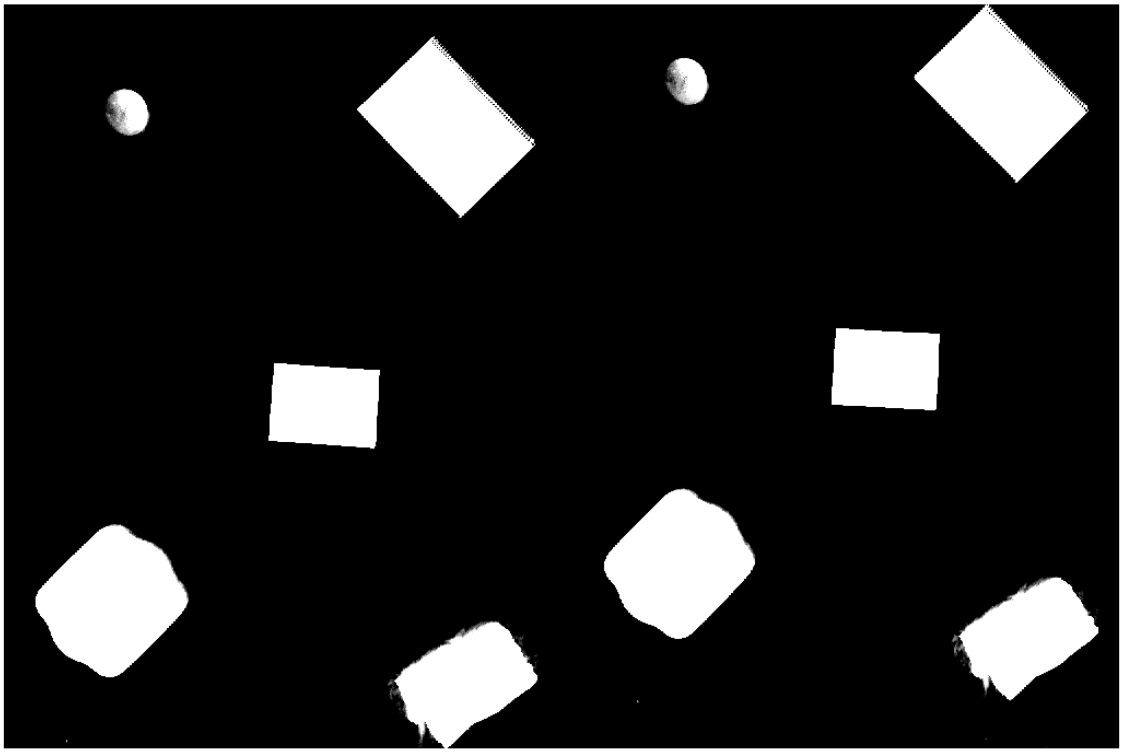


4- Negative’si Alma

Not: Klasik arka plan siyah ve cisimler beyaz olması için bu işlem gerçekleştirildi.

```
im1 = uint8(255) - imageViewer_or; % maskeleme işlemi
yaparken logic seviyesinde beyazları 1 siyahları 0 kabul
etmek için reverse aldım. imcomplement()fonksiyonu da
kullanılabilirdi.
im2 = uint8(255) - imageViewer_shif;
%şöyle de olurdu im2= ~imageViewer_shift_img_greyAdj

imshowpair(im1, im2, "montage")
```



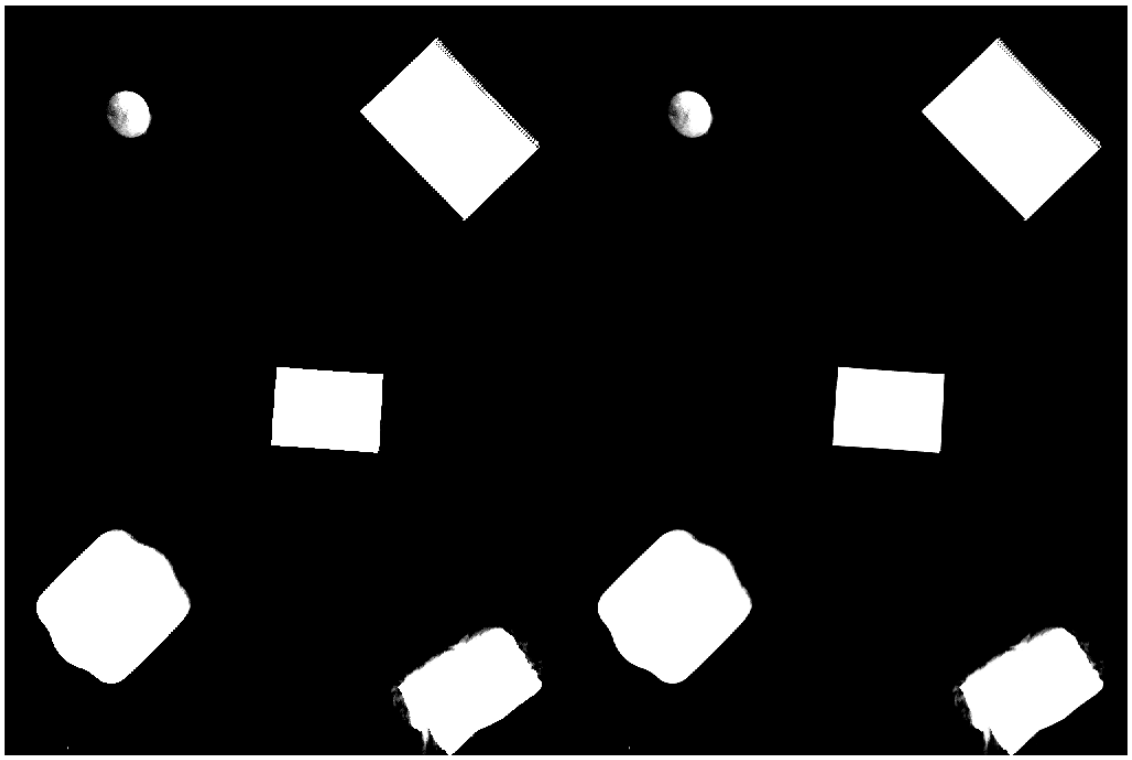
5- Filtreleme & Morfolojik işlemler

Not: Filtreleme hiç gerek yoktu ama ileride sorunlarla karşılamamak için yine de kullandım. Ayrıca kalabalık olmaması adına burada sadece or_img.jpeg fotoğrafı gösterdim. Dilation’a ise gerek duyulmadı.

5.1- Filtreleme

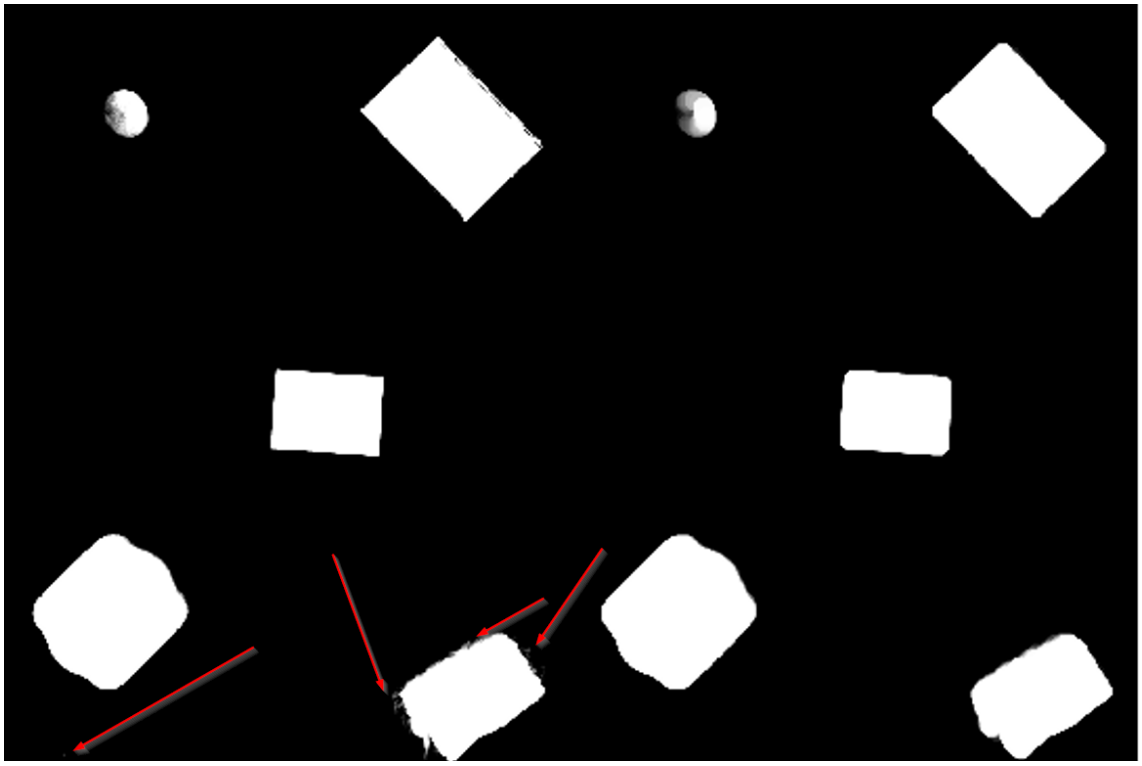
```
filter2 = fspecial("average", 3)% average 3x3

%filtreleme
org_clean = imfilter(im1, filter2);
shif_clean = imfilter(im2, filter2);
imshowpair(im1, org_clean, "montage")
imshowpair(im2, shif_clean, "montage")
```



5.2 - Erosion

```
%erosion
erosion = strel("disk", 50); % bu kadar büyük genişlikte
seçmemin sebebi kumaşın kenarlarını birer obje olarak ele
alması
c1 = imopen(org_clean, erosion);
c2 = imopen(shif_clean, erosion);
imshowpair(org_clean, c1, "montage")
imshowpair(shif_clearn, c2, “montage”)
```

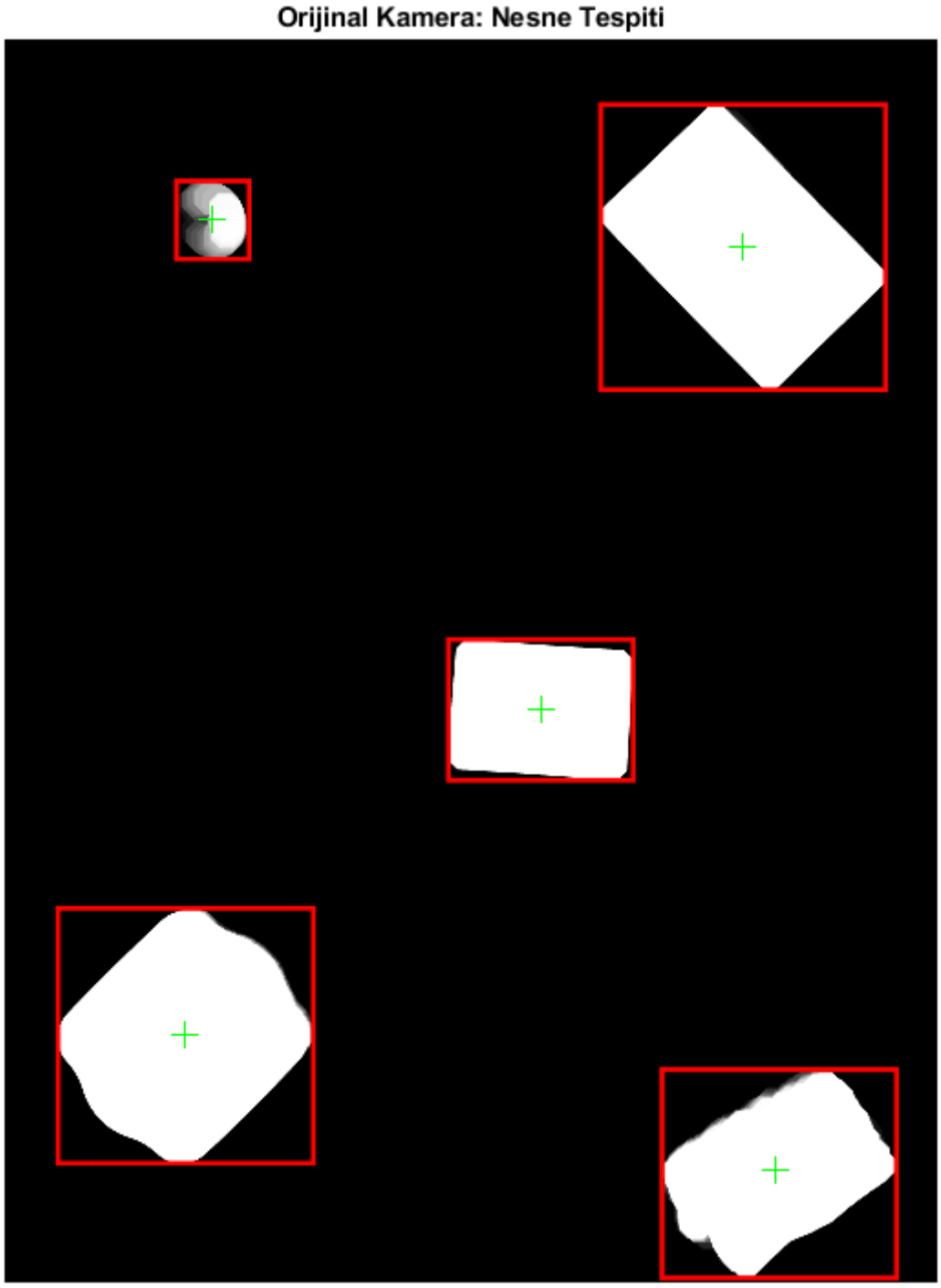


Not : Erosion sayesinde kaybolan beyaz noktalar kırmızı oklarla gösterilmiştir.

6- Centroid & Bounding Box'ları bulma

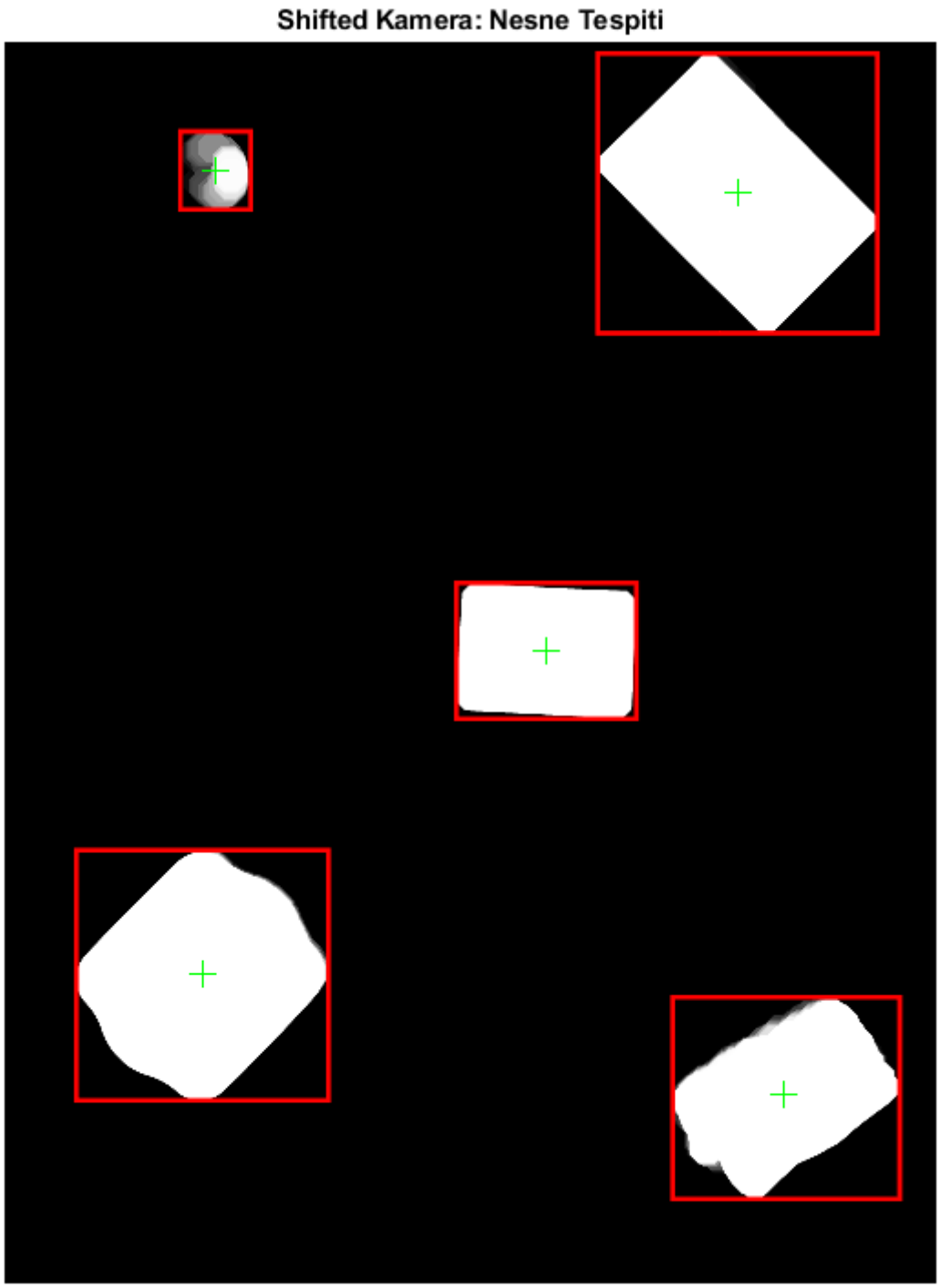
```
left_cc = bwconncomp(c1);%finds and counts the connected
components CC in the binary image BW.
right_cc = bwconncomp(c2);%finds and counts the connected
components CC in the binary image BW.

% Orijinal görüntü için çizim
left_stats = regionprops(left_cc, 'BoundingBox',
'Centroid', 'Area');
figure; imshow(c1); hold on;
for i = 1:length(left_stats)
    bbox = left_stats(i).BoundingBox;
    centroid = left_stats(i).Centroid;
    rectangle('Position', bbox, 'EdgeColor', 'r',
'LineWidth', 2);
    plot(centroid(1), centroid(2), 'g+', 'MarkerSize', 10);
end
title('Orijinal Kamera: Nesne Tespiti');
```




```
% Shifted görüntü için çizim
right_stats = regionprops(right_cc, 'BoundingBox',
'Centroid', 'Area');
figure; imshow(c2); hold on;
for i = 1:length(right_stats)
    bbox = right_stats(i).BoundingBox;
    centroid = right_stats(i).Centroid;
```

```
    rectangle('Position', bbox, 'EdgeColor', 'r',
'LineWidth', 2);
    plot(centroid(1), centroid(2), 'g+', 'MarkerSize', 10);
end
title('Shifted Kamera: Nesne Tespiti')
```



7-Elips & alan özelliklerini edinme

```
% Dosya oluştur
fid = fopen('outputs/elips_alan_ozellikleri.txt', 'w');

% Orijinal görüntü üzerine çizim ve hesaplama
figure; imshow(c1); hold on;
for i = 1:length(left_stats)
    centroid = left_stats(i).Centroid;
    area = left_stats(i).Area;
    stats_ellipse = regionprops(left_cc, 'Orientation',
'MajorAxisLength', 'MinorAxisLength');

    % Elips parametreleri
    a = stats_ellipse(i).MajorAxisLength / 2;
    b = stats_ellipse(i).MinorAxisLength / 2;
    theta = deg2rad(-stats_ellipse(i).Orientation);
    t = linspace(0, 2*pi, 100);
    x = a * cos(t);
    y = b * sin(t);
    R = [cos(theta), -sin(theta); sin(theta), cos(theta)];
    ellipse = R * [x; y];
    plot(centroid(1) + ellipse(1,:), centroid(2) +
ellipse(2,:), 'g', 'LineWidth', 1.5);

    % Alan ve elips bilgilerini yaz
    fprintf(fid, 'Obje %d:\n', i);
    fprintf(fid, ' - Alan: %.2f px^2\n', area);
    fprintf(fid, ' - MajorAxis: %.2f px\n', 2*a);
    fprintf(fid, ' - MinorAxis: %.2f px\n', 2*b);
    fprintf(fid, ' - Orientation: %.2f derece\n\n', -theta
* (180/pi));
end
title('Elipsler ve Alan Özellikleri');
saveas(gcf, 'outputs/original_ellipses.png');
fclose(fid);
```

Çıktı :

```
Obje 1:
 - Alan: 454947.00 px^2
 - MajorAxis: 817.22 px
 - MinorAxis: 719.18 px
 - Orientation: 44.84 derece

Obje 2:
 - Alan: 47950.00 px^2
 - MajorAxis: 270.88 px
```

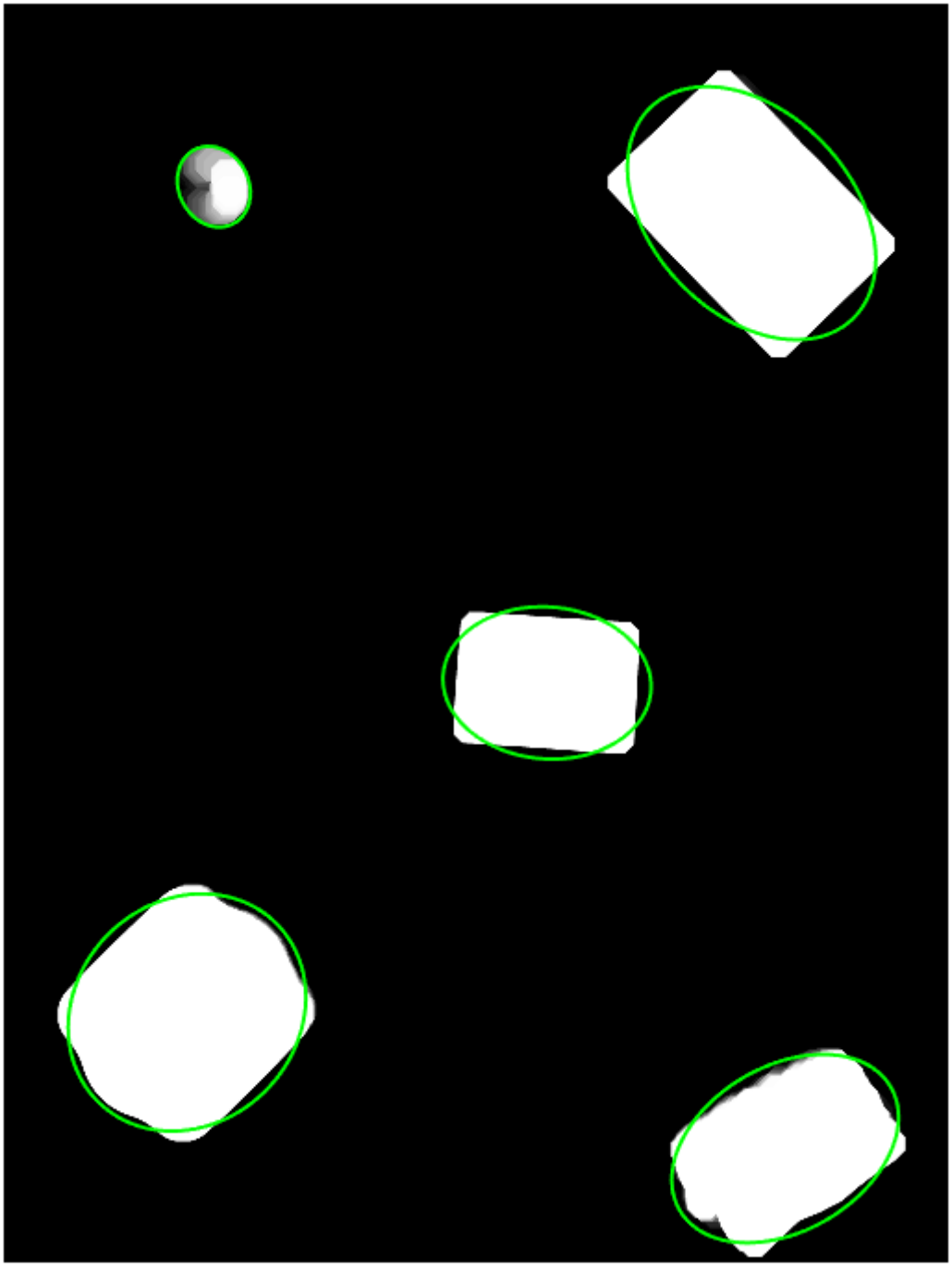
```
 - MinorAxis: 226.36 px
 - Orientation: -61.92 derece

Obje 3:
 - Alan: 250878.00 px^2
 - MajorAxis: 675.42 px
 - MinorAxis: 492.50 px
 - Orientation: -4.40 derece

Obje 4:
 - Alan: 463270.00 px^2
 - MajorAxis: 948.17 px
 - MinorAxis: 649.66 px
 - Orientation: -46.54 derece

Obje 5:
 - Alan: 325066.00 px^2
 - MajorAxis: 792.28 px
 - MinorAxis: 534.14 px
 - Orientation: 30.39 derece
```

Elipsler ve Alan Özellikleri



8-Centroid'lerinin kaymalarını hesaplama

```
% Her objenin centroid farkını hesapla
fid = fopen('outputs/centroid_disparity.txt', 'w');
disparities = zeros(length(left_stats), 1);

for i = 1:length(left_stats)
    xL = left_stats(i).Centroid(1);
    xR = right_stats(i).Centroid(1);
    disparities(i) = abs(xL - xR);
    fprintf(fid, 'Obje %d - Piksel Kaymasi (Disparity):
%.2f px\n', i, disparities(i));
end
fclose(fid);
```

Çıktı :

```
Obje 1 - Piksel Kaymasi (Disparity): 55.77 px
Obje 2 - Piksel Kaymasi (Disparity): 10.97 px
Obje 3 - Piksel Kaymasi (Disparity): 20.35 px
Obje 4 - Piksel Kaymasi (Disparity): 15.06 px
Obje 5 - Piksel Kaymasi (Disparity): 25.64 px
```

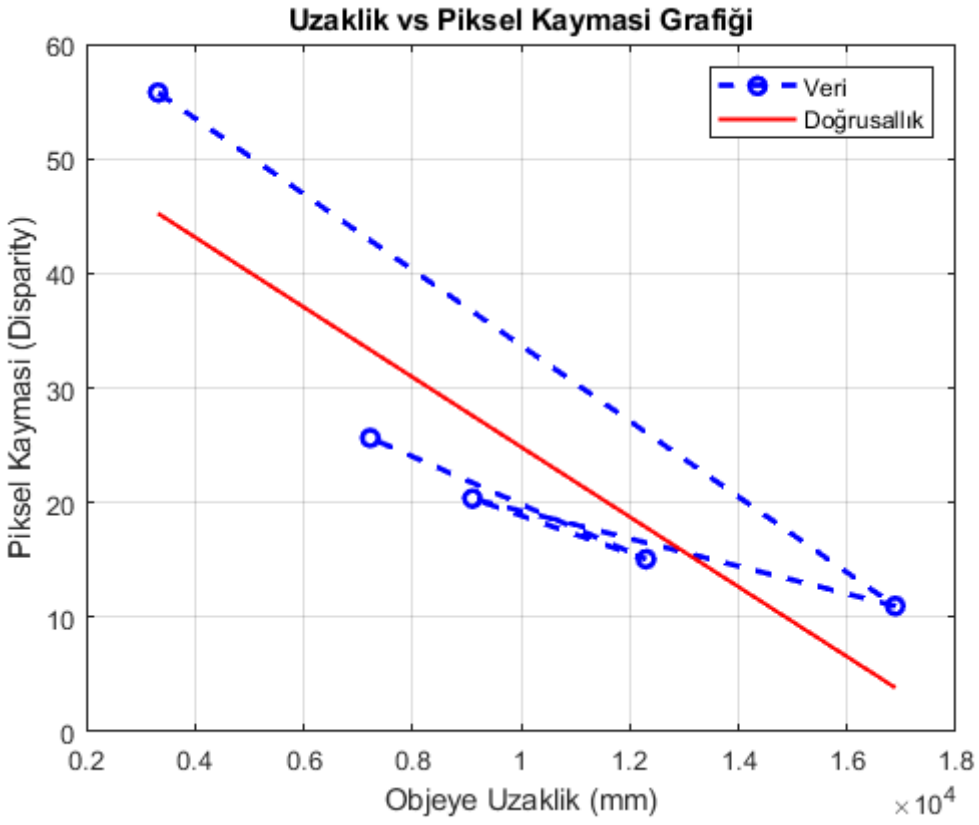
9- Uzaklık hesaplama & Grafik Oluşturma

```
% Parametreler
f = 4.15;        % Tipik cep telefonu lensi
B = 50;          % mm(5cm)
p = 0.00112;     % mm(pixel telefonun özelliğine göre
intertteki yazanlara göre varsaydım.)

% Uzaklık hesapla
depths = (f * B) ./ (disparities * p);

% Grafik çiz
figure;
plot(depths, disparities, 'bo--', 'LineWidth', 2); hold on;
xlabel('Objeye Uzaklik (mm)');
ylabel('Piksel Kaymasi (Disparity)');
title('Uzaklık vs Piksel Kaymasi Grafiği');
grid on;

% Regresyon çizgisi
pfit = polyfit(depths, disparities, 1);
plot(depths, polyval(pfit, depths), 'r-', 'LineWidth',
1.5);
legend('Veri', 'Doğrusallık');
saveas(gcf, 'outputs/graph_depth_vs_disparity.png');
```


Uzaklik vs Piksel Kaymasi Grafiği
Veri
Doğrusallık
Piksel Kaymasi (Disparity)
Objeye Uzaklik (mm)
×10^4
0
10
20
30
40
50
60
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8