



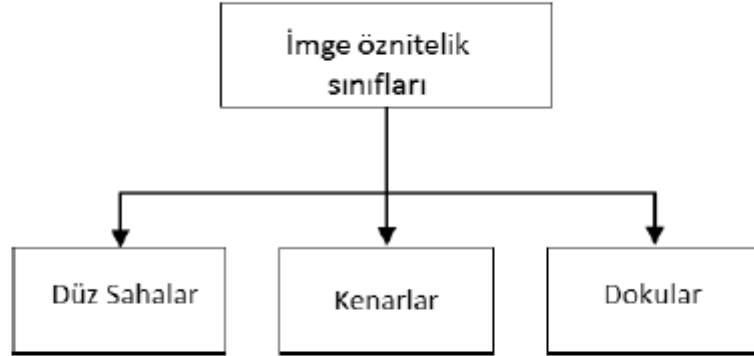
GÖRÜNTÜ İŞLEME DERS-10 YARDIMCI NOTLARI -2018

İMGE ÖZNETELİK İŞLEMLERİ:

Chen (2009)'a göre nesne tanıma için nesneyi temsil eden kesin değerlere ihtiyaç vardır. Bir yapay görme sistemi tarafından bir imge çerçevesi içinde bulunan nesnelerin tanınması ve anlamlandırılması için o nesnelere ait bir takım özellikleri (**kenar, köşe, doku, renk vs. sayısal verilere dönüştürmesi gerekir**). Bu işleme öznitelik çıkarma denilmektedir. Bu öznitelikler kimi zaman bir piksel olabildiği gibi pek çok pikselden ve aralarındaki ilişkiler yumağından oluşan değerler kümesi de olabilmektedir. Çıkarılan özniteliklerin bir nesneyi diğerlerinden ayırt edebilecek değerler taşıması gerekir. Ayrıca çıkarılan özniteliklerin sayı olarak yeterli miktarda olması gerekebilir. Bu durum bir miktar hesaplama yükünü arttıracaktır. Ancak bu şekilde sistemin hafızasında kayıtlı model ile eşleştirme işlemi doğru olarak yapılabilir. Doğru eşleştirmeler için çıkartılan özniteliklerin konumdan, dönmeden ve ölçekten bağımsız yani değişmez olmaları büyük önem taşır. Eşleştirilecek imge çiftleri farklı görüş açılarından görüntülenmiş, farklı çözünürlüklerde ya da farklı ölçeklerde alınmış olabilir. Böyle durumlarda öznitelikler değişmez yapıda olmadıkları takdirde iki imge arasındaki eşleştirme başarısız olacaktır.

Bu nedenle öznitelik çıkarma işleminde değişmez imge bilgileri kullanılarak dönme, konum ve ölçekten bağımsız değerler elde etmek gerekir. Ölçekten bağımsız yöntemler arasında SIFT (Lowe 1999) ve SURF (Bay et al. 2006) yöntemleri sayılabilir.

İmge içindeki düz sahalar imge içeriğinde en çok yer kaplayan alanlardır. İmgeyi daha uzaktan görüntülendikçe detayları kaybolarak düz sahaya dönüşür. Düz sahalar genelde tek bir gri seviye değeri ile temsil edilirler ve öznitelik çıkarmak için kısıtlı alanlardır. Düz sahaların aksine kenarlar imge içeriğinde en az yeri kaplarlar ancak imge hakkında pek çok bilgi içerirler. Kenar çizgileri sayesinde orijinal imge hakkında rahatlıkla çıkarsamalar yapılabilmektedir. İmgelerin içeriğinde bulunan nesnelere imgenin dokusunu oluşturur. Bu dokular her ne kadar gürültü içerse de aslında tanınabilecek örüntüler içerir. Yap vd. (2009) tarafından belirtilen öznitelik sınıflarının şekilsel ifadesi Şekil 2.4'de gösterilmiştir.



Şekil 2.4. Üç önemli imge sınıfı.

Çetin (2011) tarafından belirtilen tanımlamada öznitelikler global ve bölgesel öznitelikler olarak ikiye ayrılmıştır. Global öznitelikler imge içindeki nesnelerin renk yapıları, renk dağılımları, dokuları ve nesnelerin merkezleri gibi imgelerin genelini kapsayan özellikler olarak belirtilmişken bölgesel öznitelikler kenarlar, köşeler, kenarların eğrilikleri ve merkeze uzaklıklar gibi özellikler olarak belirtilmiştir. Çıkarılacak özniteliklerin global veya bölgesel seçilmesinde mevcut koşullar önem taşır. Örneğin, imge üzerindeki bir piksel öbeğinin insan yüzü olup olmadığına karar verilmesi için global özniteliklerden faydalanmak gerekirken yüz tanıma yapılmak isteniyorsa bölgesel özniteliklerden faydalanmak doğrudur.

Türkoğlu (1996) 'nın yaklaşımına göre öznitelikler genel, yapısal ve ilişkisel öznitelikler olarak üçe ayrılmıştır. Genel öznitelikler nesnelerin çevre uzunlukları, alanları, eylemsizlik momentleri ve fourier dönüşüm özellikleri gibi bilgileridir. Bu öznitelikler kolay ve hızlı bir şekilde çıkartılıp vektörel olarak düzenlenebilmektedir. Ancak bu öznitelikler çıkarılırken bir takım kısıtlar bulunmaktadır. Bunlar:

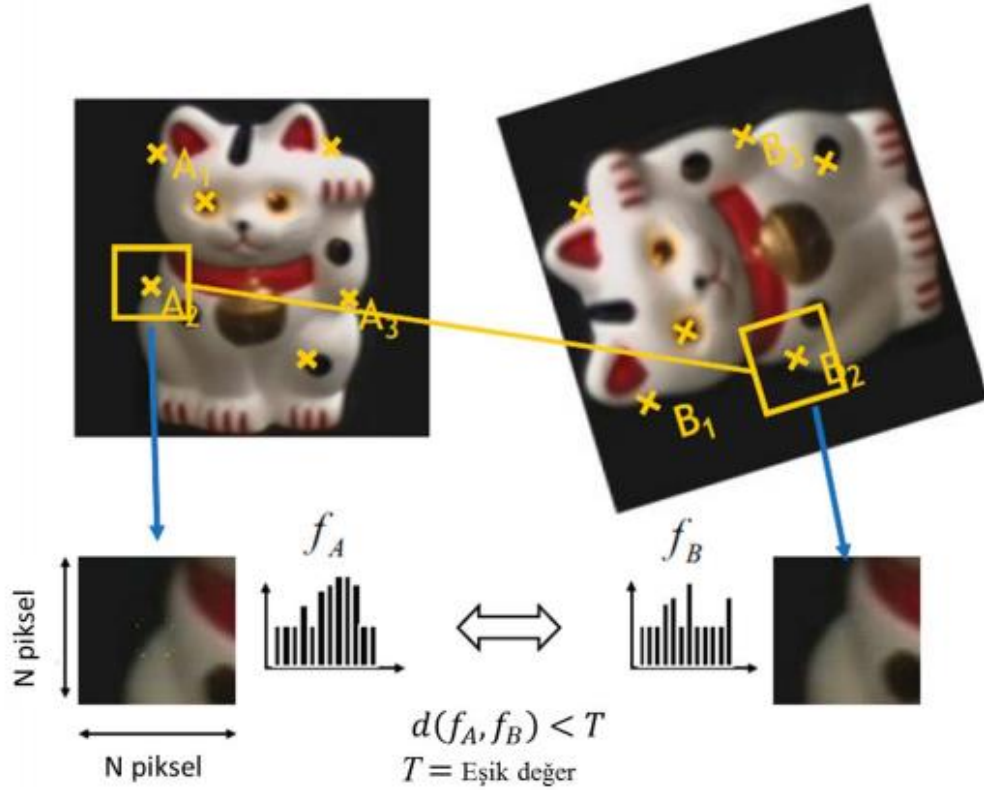
- Görüş açısına göre bağımsız değildir,
- Nesneler bir birini örtmemelidir ve şekilsel bozukluk içermemelidir,
- Bir imgedeki tüm nesnelerin ayrıştırılması için tek bir eşik değeri kullanılmalıdır.

Yapısal öznitelikler, **doğru parçası**, **eğri parçası**, **köşe** ve **kenar** gibi nesne sınırlarını oluşturacak piksel bilgilerinden oluşur. İlişkisel öznitelikler ise **kenar**, **köşe gibi özniteliklerin geometrik ilişkilerini** dikkate alır.

Grauman ve Leibe (2011) değişmez öznitelik yapılarının amaçlarının imgelerin üzerindeki yerel yapıların eşleştirilebilmesi için bir temsil sağlama olduğunu belirtmiştir. Yazarlar, güçlü öznitelik yapılarının ortaya çıkarılabilmesi için öznitelik çıkarma operatörlerinde bulunması gerekli iki önemli ölçüt belirtmişlerdir.

- Öznitelik çıkarma işlemi kesin ve tekrarlanabilir olmalıdır. Ancak bu şekilde iki ayrı imgeden çıkarılan aynı öznitelikler iki ayrı imgede de aynı nesneyi işaret edebilir.
- Özniteliklerin kendine özgü olması gerekir. Ancak bu şekilde farklı imge yapıları birbirinden ayırt edilebilir.

Yazarlar tarafından sıralanan yeterli sayıda ve doğru öznitelikler çıkarılabilmesi için gerekli işlem adımları aşağıda belirtilmiş ve Şekil 2.5’de ifade edilmiştir.



Şekil 2.5 Yerel öznitelik yapılarıyla nesne tanıma prosedürü (Grauman and Leibe 2011).

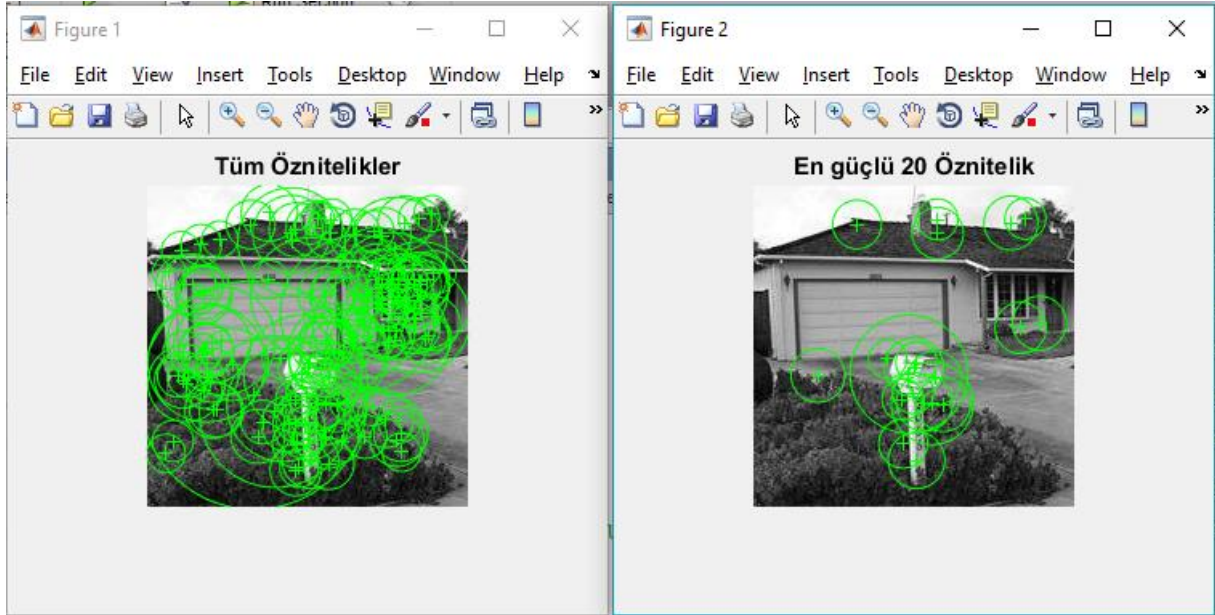
Örnek:

```
resim3='cameraman.tif';  
r3=imread(resim3);  
r3points=detectSURFFeatures(r3);  
r3fet=extractFeatures(r3,r3points);  
r3pointsLoc=r3points.Location; %r3points metotları  
gösterilebilir.  
imshow(resim3)  
hold on  
plot(r3points)  
%Çok sayıda özneliğin ekranı kaplamaması için  
plot(r3points.selectStrongest(10))
```



Farklı bir resim:

```
clear; close all;  
r=imread('Headquarters-2.jpg');  
r=rgb2gray(r);  
% SURF metoduna göre öznitelik yapısı oluşturuldu.  
rPoints=detectSURFFeatures(r);  
%Öznitelik vektörü oluşturuldu ([1x64])  
rFet=extractFeatures(r,rPoints);  
%Özniteliklerin tespit edildiği resim koordinatları bulundu.  
rPointsLoc=rPoints.Location;  
figure;imshow(r);hold on;  
%tüm öznitelikler resim üzerinde çizdiriliyor.  
plot(rPoints);  
title('Tüm Öznitelikler')  
% en güçlü 20 öznitelik resim üzerinde çizdiriliyor.  
figure;imshow(r);hold on;  
plot(rPoints.selectStrongest(20));  
title('En güçlü 20 Öznitelik')
```



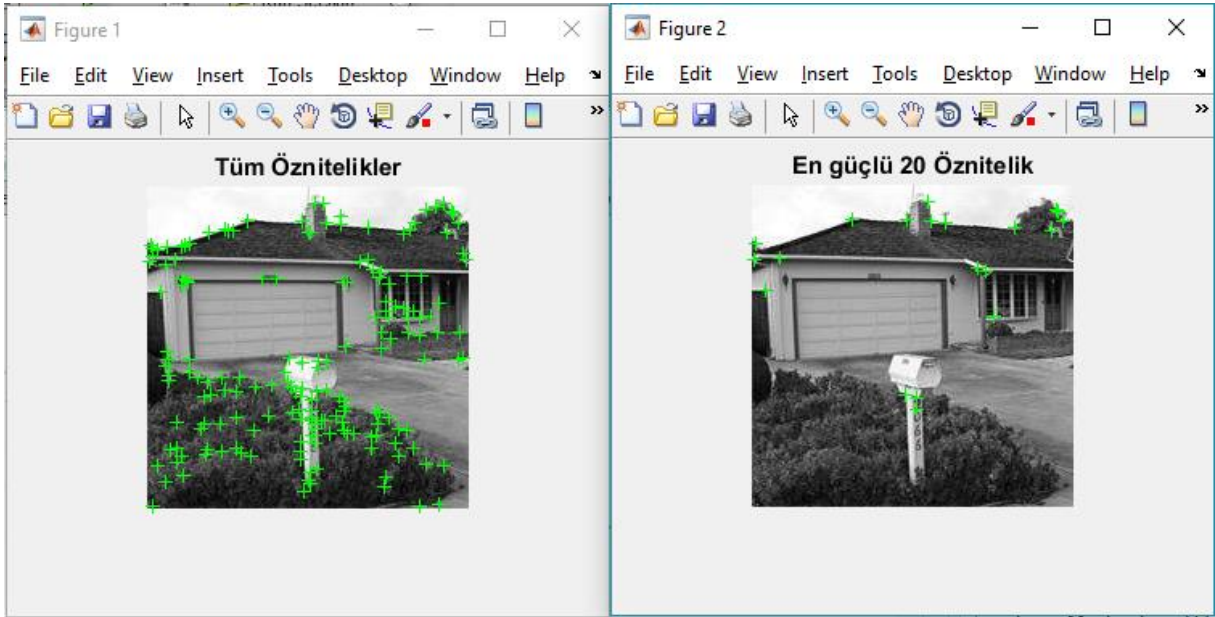
Farklı öznitelik çıkartma algoritmaları (Harris Metodu)

Harris metodu

```
clear; close all;  
r=imread('Headquarters-2.jpg');  
r=rgb2gray(r);  
% SURF metoduna göre öznitelik yapısı oluşturuldu.  
rPoints=detectHarrisFeatures(r);  
%Öznitelik vektörü oluşturuldu ([1x64])  
rFet=extractFeatures(r,rPoints);  
%özniteliklerin tespit edildiği resim koordinatları bulundu.
```



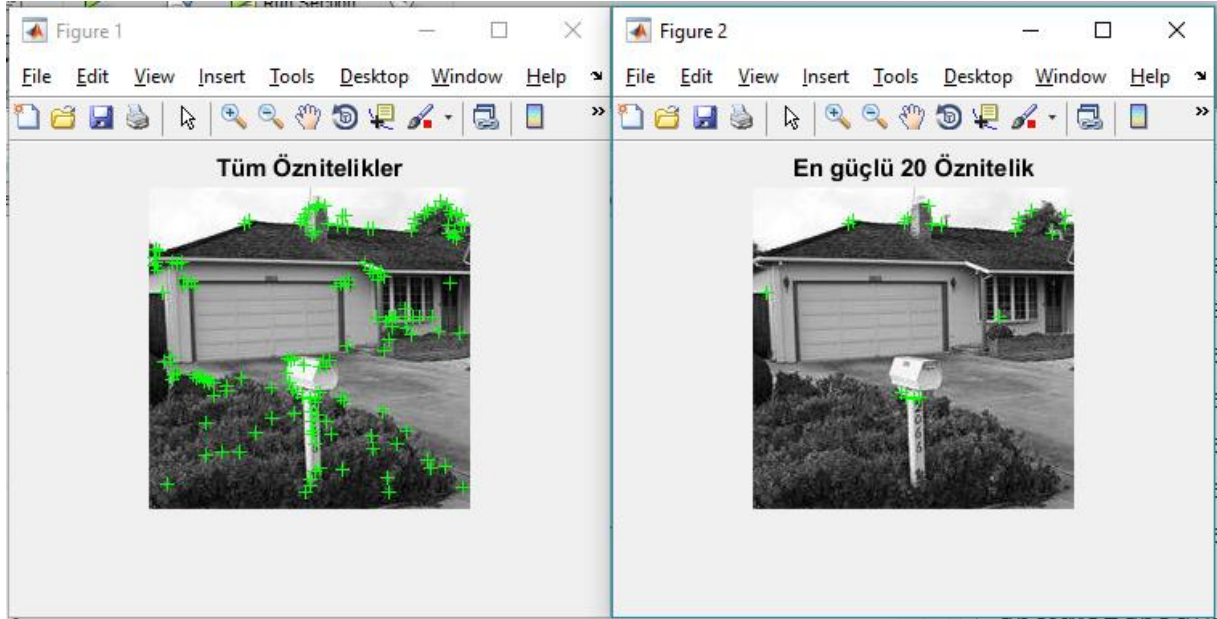
```
rPointsLoc=rPoints.Location;  
figure;imshow(r);hold on;  
%tüm öznitelikler resim üzerinde çizdiriliyor.  
plot(rPoints);  
title('Tüm Öznitelikler')  
% en güçlü 20 öznitelik resim üzerinde çizdiriliyor.  
figure;imshow(r);hold on;  
plot(rPoints.selectStrongest(20));  
title('En güçlü 20 Öznitelik')
```



Farklı öznitelik çıkartma algoritmaları (FAST Metodu)

FAST metodu

```
clear; close all;  
r=imread('Headquarters-2.jpg');  
r=rgb2gray(r);  
% SURF metoduna göre öznitelik yapısı oluşturuldu.  
rPoints=detectFASTFeatures(r);  
%Öznitelik vektörü oluşturuldu ([1x64])  
rFet=extractFeatures(r,rPoints);  
%özniteliklerin tespit edildiği resim koordinatları bulundu.  
rPointsLoc=rPoints.Location;  
figure;imshow(r);hold on;  
%tüm öznitelikler resim üzerinde çizdiriliyor.  
plot(rPoints);  
title('Tüm Öznitelikler')  
% en güçlü 20 öznitelik resim üzerinde çizdiriliyor.  
figure;imshow(r);hold on;  
plot(rPoints.selectStrongest(20));  
title('En güçlü 20 Öznitelik')
```



(Surf, Harris ve Fast metodlarının öznitelik sayılarını ve öznitelik vektörleri içindeki değerleri karşılaştırınız. Yöntemlerin güçlü ve zayıf yönlerini araştırınız.)

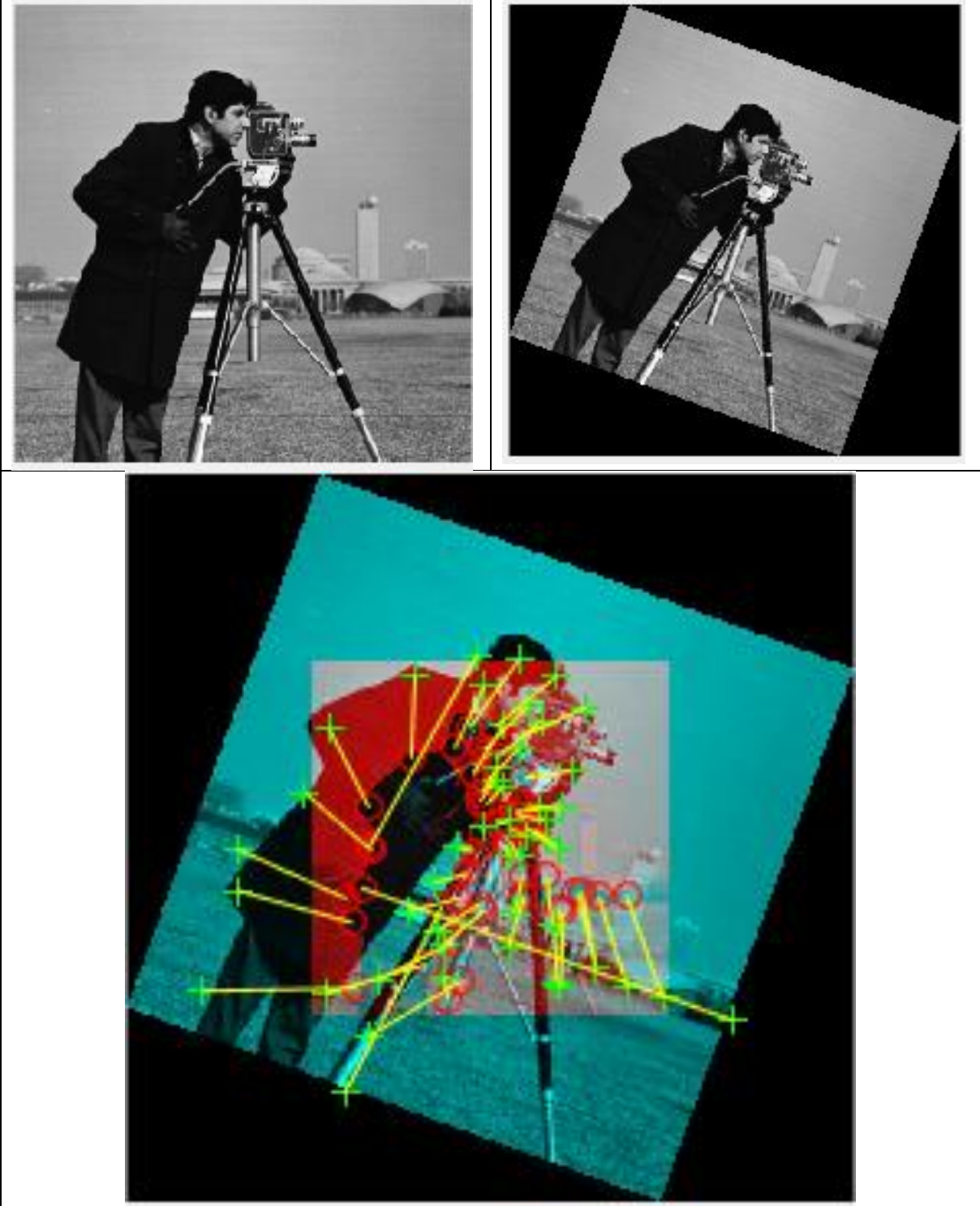
ÖZNİTELİK EŞLEŞTİRME:

```
clear; close all;
I1 = imread('cameraman.tif');
% 2. resim boyutları değiştiriliyor.
I2= imresize(I1,1.6);
% 2. resim -20 derece döndürülüyor.
% I2 = imrotate (I2,-20);
figure, imshow(I1);
figure, imshow(I2);
% her iki resim için öznitelik yapıları oluşturuldu.
points1=detectSURFFeatures(I1);
points2=detectSURFFeatures(I2);
% her iki resim için öznitelik vektörleri ve öznitelik
koordinatlarını oluşturuyoruz.
[f1, loc1]=extractFeatures(I1,points1);
[f2, loc2]=extractFeatures(I2,points2);
% 2 resimdeki eşleşen öznitelikler çiftler halinde bulunuyor.
indexPairs=matchFeatures(f1,f2);
% her iki resimde eşleşen özniteliklerin
% koordinatları bulunuyor.
matchedPoints1=loc1(indexPairs(:,1));
matchedPoints2=loc2(indexPairs(:,2));
%görüntüleme yapılıyor.
```



figure;
showMatchedFeatures (I1, I2, matchedPoints1, matchedPoints2);

Program çıktıları:

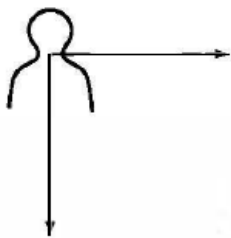
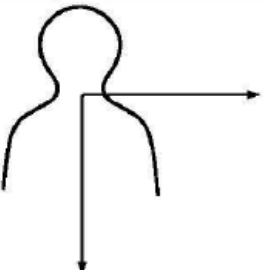
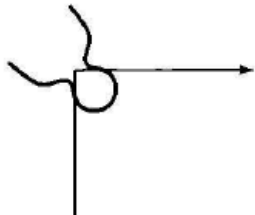




GEOMETRİK DÖNÜŞÜM:

Aşağıdaki tablolarda verilen matrisler kullanılarak imgeler “Digram” kısmında verildiği şekilde geometrik olarak döndürülebilirler.

TABLE 6.1 Types of affine transformations.

Type	Affine Matrix, T	Coordinate Equations	Diagram
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= w \\ y &= z \end{aligned}$	
Scaling	$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= s_x w \\ y &= s_y z \end{aligned}$	
Rotation	$\begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= w \cos \theta - z \sin \theta \\ y &= w \sin \theta + z \cos \theta \end{aligned}$	



Shear (horizontal)	$\begin{bmatrix} 1 & 0 & 0 \\ \alpha & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= w + \alpha z \\ y &= z \end{aligned}$	
Shear (vertical)	$\begin{bmatrix} 1 & \beta & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= w \\ y &= \beta w + z \end{aligned}$	
Vertical reflection	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= w \\ y &= -z \end{aligned}$	
Vertical reflection	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= w \\ y &= -z \end{aligned}$	
Translation	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \delta_x & \delta_y & 1 \end{bmatrix}$	$\begin{aligned} x &= w + \delta_x \\ y &= z + \delta_y \end{aligned}$	

Matlab Geometrik Dönüşüm İşlemleri

```
clear; close all;  
r=imread('Headquarters-2.jpg');  
% Ölçekleme geometrik dönüşüm matrisi oluşturuluyor.  
tformOlcekle = affine2d([1.5 0 0; 0 1.1 0; 0 0 1]);  
% dönüştürülmüş görüntü elde ediliyor.  
q=imwarp(r,tformOlcekle);
```



```
figure, imshow(r);  
title('Orjinal Görüntü')  
figure, imshow(q);  
title('x ve y eksenlerinde büyütülmüş görüntü')  
  
% Açısal döndürme geometrik dönüşüm matrisi oluşturuluyor.  
tformAcisal = affine2d([0.866 0.5 0; -0.5 0.866 0; 0 0 1]);  
q2=imwarp(r,tformAcisal);  
figure, imshow(q2)  
title('30 derece açıyla döndürülmüş görüntü')  
  
% Horizontal Shear geometrik dönüşüm matrisi oluşturuluyor.  
tformHorShear = affine2d([1 0 0; 1.3 1 0; 0 0 1]);  
q3=imwarp(r,tformHorShear);  
figure, imshow(q3);  
title('Horizontal Shear görüntü')
```

Dönüşüm Matrisi (Ölçekleme)

$$\begin{bmatrix} 1.50 & 0 & 0 \\ 0 & 1.10 & 0 \\ 0 & 0 & 1.00 \end{bmatrix}$$

Dönüşüm Matrisi (Açısal döndürme)

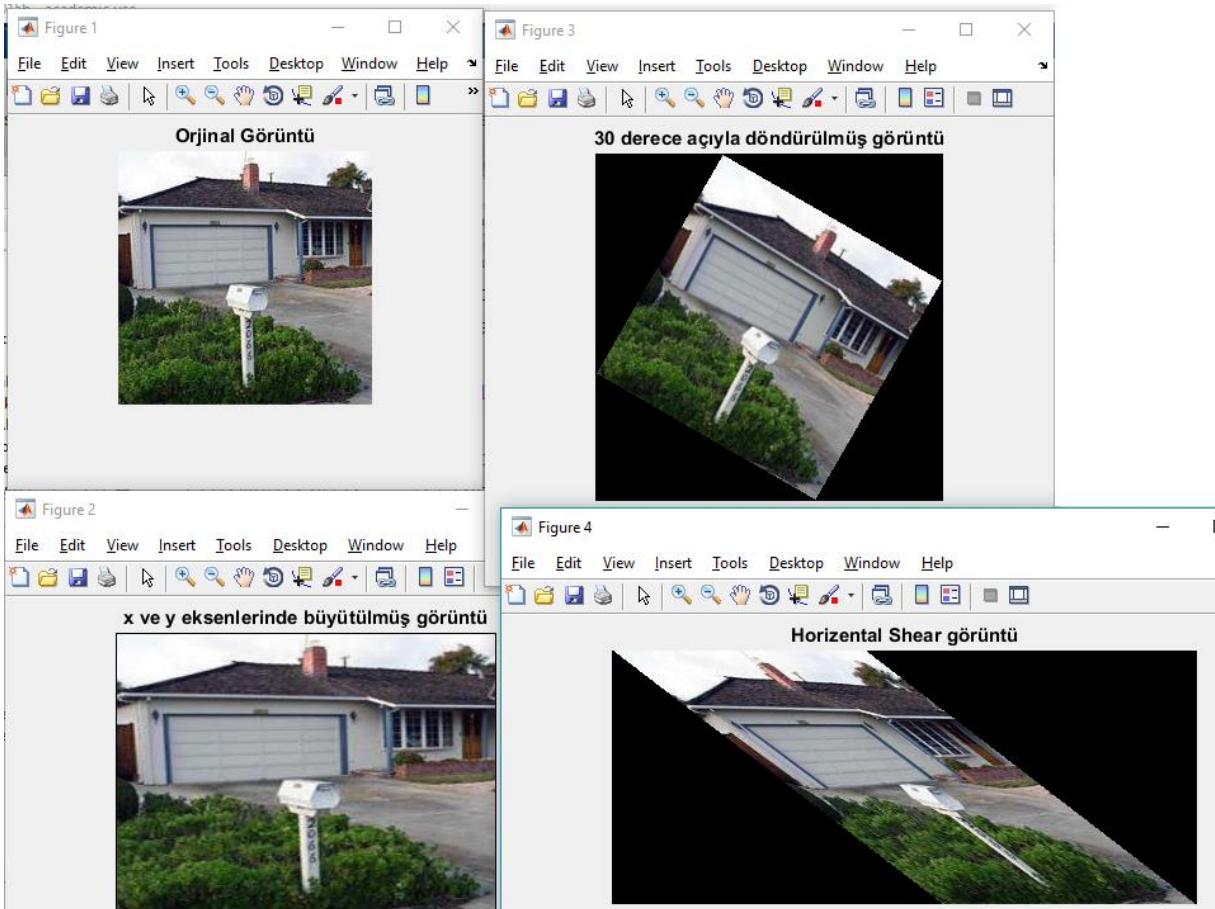
$$\begin{bmatrix} 0.87 & 0.50 & 0 \\ -0.50 & 0.87 & 0 \\ 0 & 0 & 1.00 \end{bmatrix}$$



Dönüşüm Matrisi (Horizontal Shear)

$$\begin{bmatrix} 1.00 & 0 & 0 \\ 1.30 & 1.00 & 0 \\ 0 & 0 & 1.00 \end{bmatrix}$$

Program çıktıları:





GÖRÜNTÜ İŞLEME TEKNİKLERİYLE NESNE TESPİT ETME ve SAYMA

Tek bir görüntüde ya da bir video akışında bulunan nesnelerin çeşitli algoritma ve yazılımlar desteğiyle bilgisayar sistemleri tarafından tespit edilip anlamlandırılması olarak tanımlanabilir. Bu tespit işlemi; görüntü üzerindeki tek bir nesnenin tespit edilip işlenmesi ve anlamlandırılması şeklinde (plaka tanıma, yüz tanıma vb.) olduğu gibi görüntü akışında hareket eden bir nesnenin uzamsal yer değiştirmesi, davranış şekli, geometrik ve şekilsel değişiklikleri de anlamlandırılabilir.

Görüntü çerçevelerindeki nesnelerin tanınması işlemlerinde ilk aşama nesnenin doğru bir şekilde tespit edilmesidir. Bu işlem için konum-zaman ilişkisi çerçevesinde bazı ayrımların yapılması gerekmektedir. Tespit edilecek nesne, ardışık çerçeveler içinde hareketsiz kalarak engel niteliği alabildiği gibi yine ardışık çerçeveler içerisinde hareketli olabilir. Ayrıca; görüntü çerçevelerinin elde edildiği kameralar sabit olabileceği gibi serbest hareketli ya da tekdüze hareketli olabilir. Bu nedenle; tespit algoritmaları oluşturulurken bu durumun dikkate alınması gerekmektedir. Juang ve Chen (2008) nesne tespit ve tanıma işlemini başlıca 3 adım olarak tarif etmiştir. Birinci adım ön plandaki nesnenin arka plandan ayrılması, ikinci adım nesnenin özniteliklerinin çıkarılması ve üçüncü adım ise nesnenin tanınması olarak ifade edilmiştir.

Nesne sayımı, görüntü işlemede zor bir sorundur. Diğer sektörler, araştırma enstitüleri, laboratuvarlar, tarım endüstrileri gibi farklı alanlarda rutin olarak yürütülür. Nesne sayımı, belirli elemanların tahminine bağlı olan nicel analiz için önemlidir. Nesne sayımı için geleneksel yöntemler manuel, zaman alıcı ve otomatik olmayan yöntemlerdir. Sürekli sayım göz yorgunluğuna yol açar ve sonuçların doğruluğunu etkiler. Otomatik sayım objektif, güvenilir ve tekrarlanabilir olduğundan, örnekler arasında hücre sayısının karşılaştırılması, otomatik programlarla manüel saymaya kıyasla çok daha doğrudur. Biyomedikal ve Tıp sektörlerinde hücre sayımı tıbbi teşhis ve biyolojik araştırma için çok önemlidir ve yararlıdır. Mikroorganizmaların ve kolonilerin sayılması, sağlık testlerinde, gıda kalite kontrolünde, tarım analizinde vb. en temel faaliyetlerden biridir. Kan sayımı, tıpta en sık uygulanan kan testlerinden biridir.

Basit bir nesne tespit ve sayma işlemi: Bir imgedeki mavi renkli nesnelerin tespit ve sayma işlemi:





AKÜ TEKNOLOJİ FAKÜLTESİ
MEKATRONİK MÜHENDİSLİĞİ BÖLÜMÜ



```
%%
clear;
close all;clc;
a=imread('Kalemler.jpg');
figure(1);imshow(a);

%%
aMavi=a(:,:,3);%Mavi renkli kanal imgeden ayrıldı.
aGri=rgb2gray(a);%gri seviye imge elde edildi.
% aKirmizi=a(:,:,1);
b=imsubtract(aMavi,aGri);%Mavi kanal, gri seviye resimden çıkartılarak en
mavi olan pikseller tespit edildi.
% bb=imsubtract(aKirmizi,aGri);%kırmızı nesnelere bulmak istersek
figure(2);imshow(b);

%%
c=b>20; %ikilik resim elde edildi.
figure(3);imshow(c);
%%
%ikilik resim üzerinde morfolojik düzeltmeler yapılıyor.
filtre=strel('square',13);
d=imclose(c,filtre);
figure(4);imshow(d);
%%
e=bwareaopen(d,100); %küçük piksel öbekleri yok edildi.
figure(5);imshow(e);
%
%%
f=imclose(e,filtre);
f=imfill(f,'holes');
figure(6);imshow(f);
%%
%imgedeki nesnelere sayılma üzere etiketlendi ve sayıları bulundu.
[etiketlenmisGoruntu, nesneSayisi]=bwlabel(f);
% etiketlenmisGoruntu=uint8(etiketlenmisGoruntu.*30);
% figure(7);imshow(etiketlenmisGoruntu);
fprintf('İmgedeki mavi renkli nesne sayısı:%d\n',nesneSayisi);
%%
%nesnelere yer tespitleri ve diğer bir kısım özellikleri bulunuyor.
hold on
nesneler=regionprops(f,'all');%imge içinde yakalanan her bir nesnenin
şekilsel özelliklerinin sayısal değerleri hesaplandı.İmge içinde kaç nesne
tespit edildiyse nesnelere dizisinin o kadar elemanı vardır. Dolayısıyla her
bir nesneye bir for döngüsü ile erişmek gerekir.
stats = regionprops('table',f,'Centroid', 'MajorAxisLength', 'MinorAxisLength')
for i=1:length(nesneler)
plot(nesneler(i).Centroid(1),nesneler(i).Centroid(2),'b*')
rectangle('Position',nesneler(i).BoundingBox,'EdgeColor','r','LineWidth',2);
text(nesneler(i).BoundingBox(1)-20,nesneler(i).BoundingBox(2)-15,
strcat(num2str(nesneler(i).BoundingBox(1)),', ',
num2str(nesneler(i).BoundingBox(2))), 'Color','Yellow');
end
```



AKÜ TEKNOLOJİ FAKÜLTESİ
MEKATRONİK MÜHENDİSLİĞİ BÖLÜMÜ



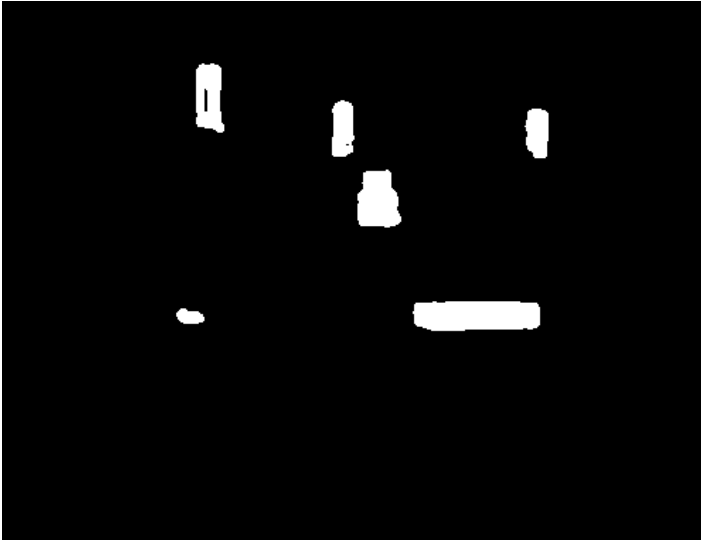
Figure(1)



Figure(2)

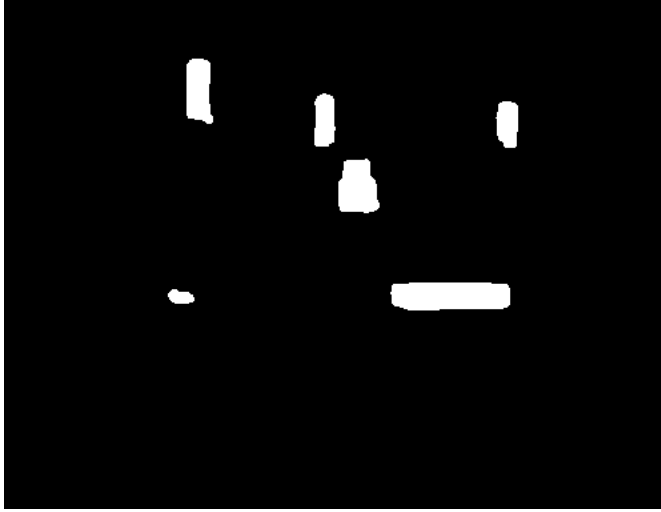


Figure(3)





Figure(4)



Figure(5)



Figure(6)

