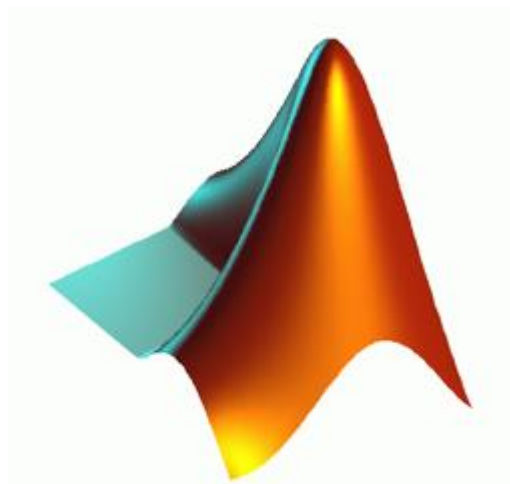


Uzaktan Algılama Teknolojileri

Ders 2 – MATLAB ve Görüntü İşleme

Alp Ertürk

alp.erturk@kocaeli.edu.tr



MATLAB

- Matrix Laboratory'nin kısaltmasıdır
- Bir çok uygulamada kolaylık sağlayacak özelleşmiş parçaları olan bir üst seviye programlama dili ve programdır
- Özellikle matris ve vektör işlemlerinde C veya Fortran gibi programlama dillerine göre çok daha kısa sürede işlemi tamamlamayı sağlar
- Temel fonksiyonlardan daha karmaşık fonksiyonlara kadar geniş bir hazır fonksiyon kütüphanesi vardır

MATLAB

Command Window

- Komut yazımı için

Current Directory

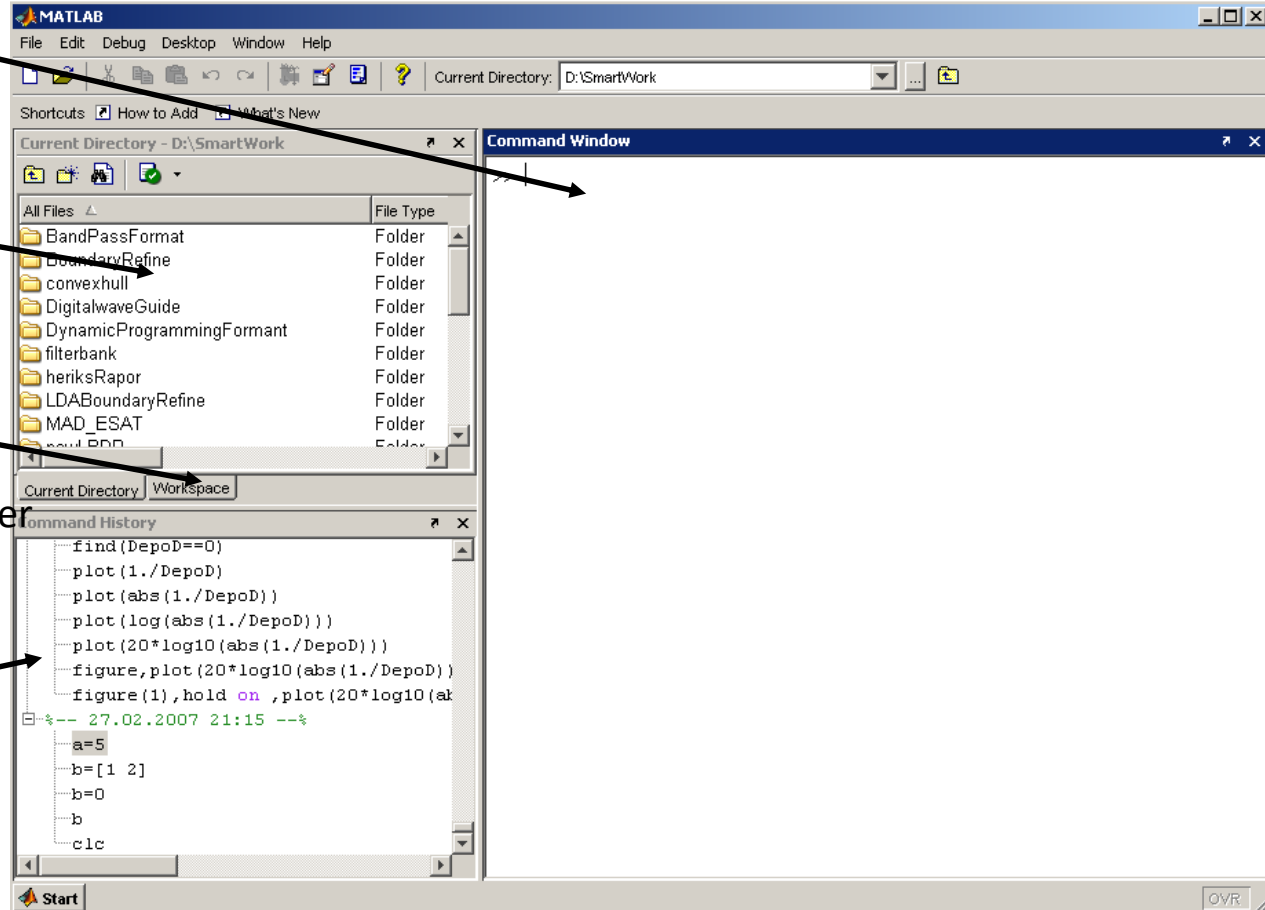
- Aktif klasör ve alt-dosyalar

Workspace

- Program değişkenleri
- Çift tıklanıldığında değişkenler Array Editor'de açılır

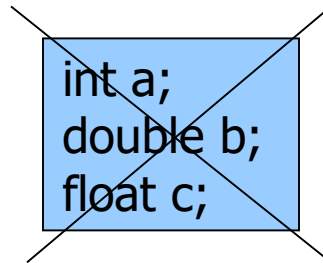
Command History

- Önceki komutları gösterir



MATLAB

- Değişken tiplerini belirtmeye gerek yoktur
- Aksi belirtilmedikçe tüm değişkenler çift hassasiyette (double precision) yaratılır



```
int a;  
double b;  
float c;
```

MATLAB

- MATLAB'da matris yaratmak için,
 - Elemanlar köşeli parantez içine alınır, []
 - Her satırın sonunu belirtmek için noktalı virgül kullanılır, ;
 - Satır içindeki elemanlar virgül ile veya boşlukla ayrılır

$A = [16 \ 3 \ 2 \ 13; 5 \ 10 \ 11 \ 8; 9 \ 6 \ 7 \ 12; 4 \ 15 \ 14 \ 1]$

MATLAB

- Değişken oluşturulması (veya komut) sonrası noktalı virgül kullanılmazsa MATLAB sonucu komut penceresinde (command window) gösterir

A =

16 3 2 13

5 10 11 8

9 6 7 12

4 15 14 1

- Oluşturulan bir değişken MATLAB iş alanında (workspace) yer alır
- Not: Değişken isimleri büyük-küçük harfe duyarlıdır

MATLAB

- Değişken ismi atanmazsa (örn. $A =$), MATLAB komut sonucunu *ans* (answer'ın kısaltılmışı) isimli geçici bir değişkende saklar
- Matrisin elemanlarına ulaşmak için $A(i, j)$ yapısı kullanılır. i alanı kaçınıcı satır olduğunu, j alanı ise kaçınıcı sütun olduğunu göstermek içindir
- Örneğin $A(3,4)$, A matrisinin 3. satırında ve 4. sütununda yer alan matris elemanına işaret etmektedir.
- $A(1,4) + A(2,4) + A(3,4) + A(4,4)$ şeklinde A 'nın 4. sütunundaki elemanları toplamı elde edilir (A dört satırlı ise, aksi halde 4. sütundaki ve ilk dört satırdaki elemanların toplamı)

MATLAB

- İki nokta üstüste operatörü:

- Belirli bir artışla birden fazla elemanı veya sayıyı almak için kullanılır
- Örneğin 1:10, 1'den 10'a kadar olan sayılardır

1 2 3 4 5 6 7 8 9 10

- 100:-7:50 şeklinde ise 100'den başlayan ve 7 azalarak 50'ye kadar giden sayılar alınır:

100 93 86 79 72 65 58 51

- A(2:4,3), A matrisinin 2. satırdan 4. satıra kadar ve 3. sütunda olan elemanlarına işaret etmektedir.

MATLAB

- Eşit aralıklı elemanlardan oluşan vektör oluşturmak:

- Aralık belirtilerek: $x = 0 : 0.5 : \pi$

$x =$

0 0.5000 1.0000 1.5000 2.0000 2.5000 3.0000

- Eleman sayısı belirtilerek: $x = \text{linspace}(0, \pi, 7)$

$x =$

0 0.5236 1.0472 1.5708 2.0944 2.6180 3.1416

- Logaritmik olarak eşit aralıklı: $x = \text{logspace}(1, 2, 7)$

$x =$

10.0000 14.6780 21.5443 ... 68.1292 100.0000

MATLAB

- Hazır matris oluşturma komutları:
 - `zeros(m,n)` : m satır ve n sütunlu, sıfırlardan oluşan matris
 - `ones(m,n)` : m satır ve n sütunlu, birlerden oluşan matris
 - `eye(n,n)` : n satır ve n sütunlu, birim matris
 - `rand(m,n)` : m satır ve n sütunlu, uniform dağılımlı rastgele değerlerden oluşan matris
 - `randn(m,n)` : m satır ve n sütunlu, uniform dağılımlı rastgele değerlerden oluşan matris

MATLAB

- Matris işlemleri:
 - $+$: Toplama
 - $-$: Çıkarma
 - $*$: Çarpma
 - $/$: Bölme
 - \backslash : Sol taraftaki matrisin tersi ile sağ taraftaki matrisi çarpma
 - $'$: Devriğini alma
 - $^$: Üst alma (Kare matrislerde tanımlı)

MATLAB

- Eleman – elemana matris işlemleri:
 - `.*` : Eleman elemana çarpma
 - `./` : Eleman elemana bölme
 - `.^` : Elemanların üstünü alma

MATLAB

- Hazır temel vektör fonksiyonları:
 - `mean(A)` : Vektörün ortalamasını alma
 - `max(A)` , `min(A)` : Vektörün en büyük ve en küçük değerlerini alma
 - `sum(A)` : Vektör elemanlarının toplamı
 - `median(A)` : Vektörün ortanca değeri
 - `std(A)` : Vektörün standart sapması
 - `det(A)` : Kare matrisin determinantı
 - `dot(A,B)` : İki vektörün nokta çarpımı
 - `cross(A,B)` : İki vektörün çapraz çarpımı
 - `inv(A)` : Matris tersi
 - `size(A)` : Matris veya vektörün boyutları
 - ...
 - ...

MATLAB

- Mantıksal / ilişkisel operatörler:
 - == : Eşittir (Atama işlemi ile farklıdır)
 - ~= : Eşit değildir
 - < : Küçüktür
 - > : Büyüktür
 - <= : Küçük eşittir
 - >= : Büyük eşittir
 - & : 've' operatörü
 - | : 'veya' operatörü

MATLAB

- Akış kontrolü operatörleri
 - if
 - for
 - while
 - switch
 - break

MATLAB

- if koşulu:

```
if (koşul_1)
    MATLAB komutları
elseif (koşul_2)
    MATLAB komutları
elseif (koşul_3)
    MATLAB komutları
else
    MATLAB komutları
end
```

MATLAB

- Örn.:

```
if (a<3)
    disp('a is smaller than 3');
elseif (a==3)
    disp('a is equal to 3');
else
    disp('a is larger than 3');
end
```

MATLAB

- Örn.:

```
if ((a<3)&(b==5))
    disp('a is smaller than 3, and b is 5');
elseif (a==3)
    disp('a is equal to 3');
else
    disp('a is larger 3');
    disp('or a is not 3 and b is not 5 ');
end
```

MATLAB

- for döngüsü:

```
for i = 1:10  
    disp(i);  
end
```

```
for i = 1:2:10  
    disp(i);  
end
```

MATLAB

- for döngüsü:

```
for i = 10:-2:1
    disp(i);
end
=> 10 8 6 4 2
```

```
for i = [0.1 0.3 -1 4 7]
    disp(i);
end
=> 0.1 0.3 -1 4 7
```

MATLAB

- while döngüsü:

```
i = 1;
```

```
while (i<=10)
```

```
    disp(i);
```

```
    i = i+1;
```

```
end
```

```
=> 1 2 3 4 5 6 7 8 9 10
```

MATLAB

- Eğer y bir vektörse, `plot(y)` x-ekseninde eleman indisleri olacak şekilde y 'nin elemanlarının değerleri grafiğini çizer
- `plot(x,y)` şeklinde iki değişken kullanıldığında y vs. x grafiği çizilir
- Eksenleri etiketlemek için *xlabel* ve *ylabel* komutları, tüm grafiği etiketlemek için *title* komutu kullanılır

MATLAB

- plot komutu için renk ve çizgi stili de kullanıcı tarafından belirlenebilmektedir.
- Örneğin `plot(x,y,'r:+')` komutu, çizimi kırmızı renkte, noktalı olarak ve her veri noktasında + işareti olacak şekilde çizer
- Daha fazla bilgi için komut penceresinde “help plot”
- Yeni bir boş şekil penceresi için “figure”
- Mevcut şekil penceresinin üstüne tekrar çizim yapmak için “hold on”

MATLAB

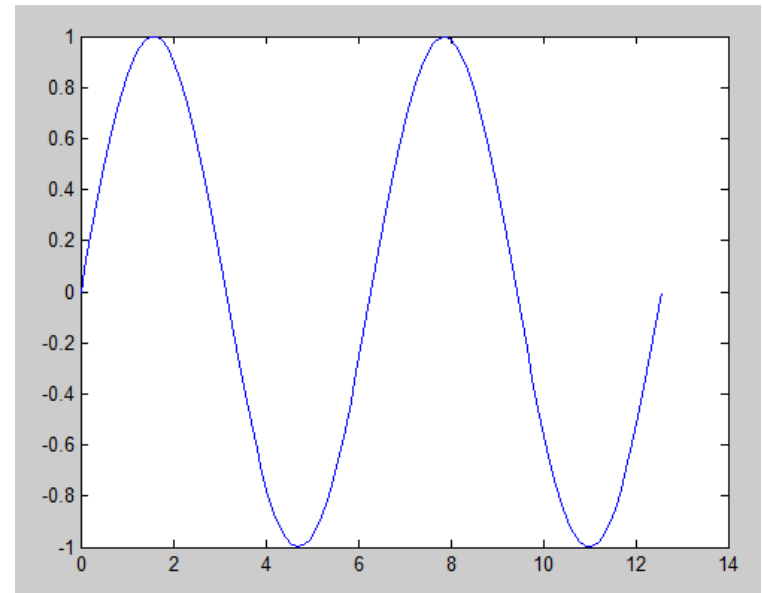
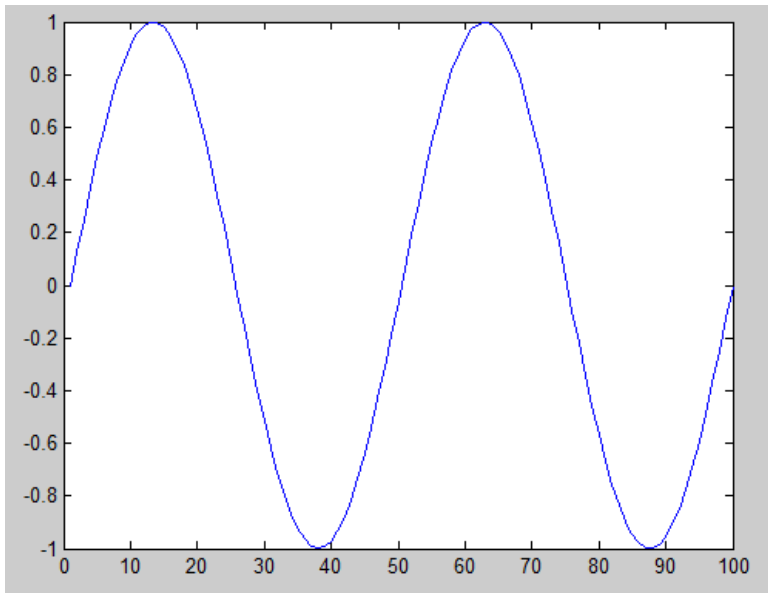
- Örn.:

```
x=linspace(0,4*pi,100);
```

```
y=sin(x);
```

```
figure; plot(y)
```

```
figure; plot(x,y)
```



MATLAB

- Tek bir plot komutuyla çoklu x - y eşli grafikleri çizilebilir

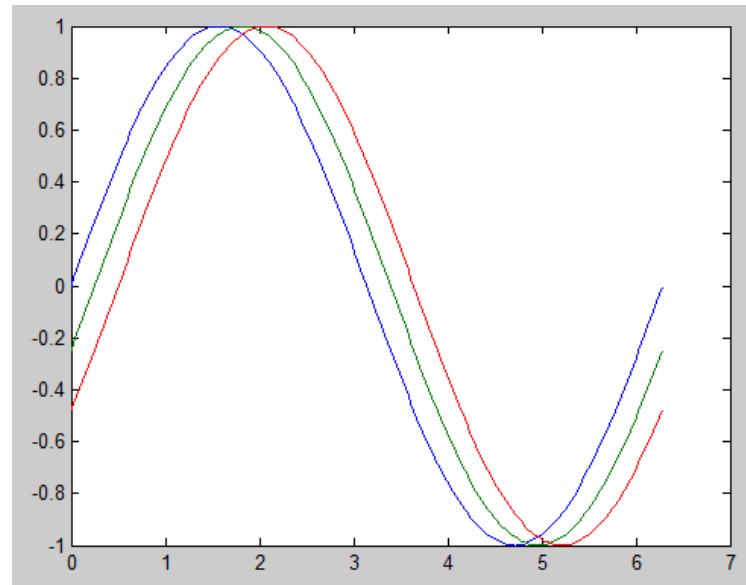
```
x = 0:pi/100:2*pi;
```

```
y = sin(x);
```

```
y2 = sin(x-.25);
```

```
y3 = sin(x-.5);
```

```
plot(x,y,x,y2,x,y3)
```



MATLAB

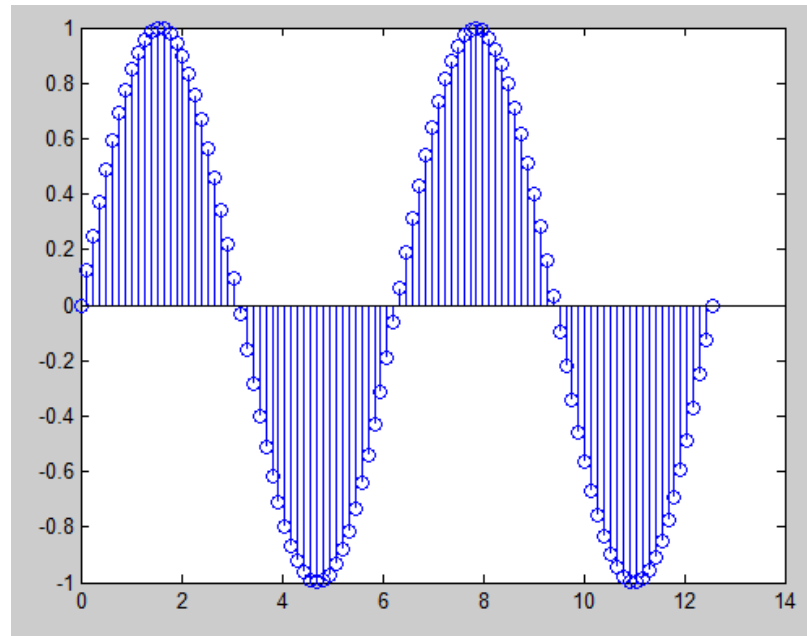
- stem komutuyla grafikler ayrık-zamanlı olarak çizilebilir

- Örn.:

```
x=linspace(0,4*pi,100);
```

```
y=sin(x);
```

```
figure; stem(x,y)
```



MATLAB

- Örn.:

```
x=linspace(0,4*pi,100);
```

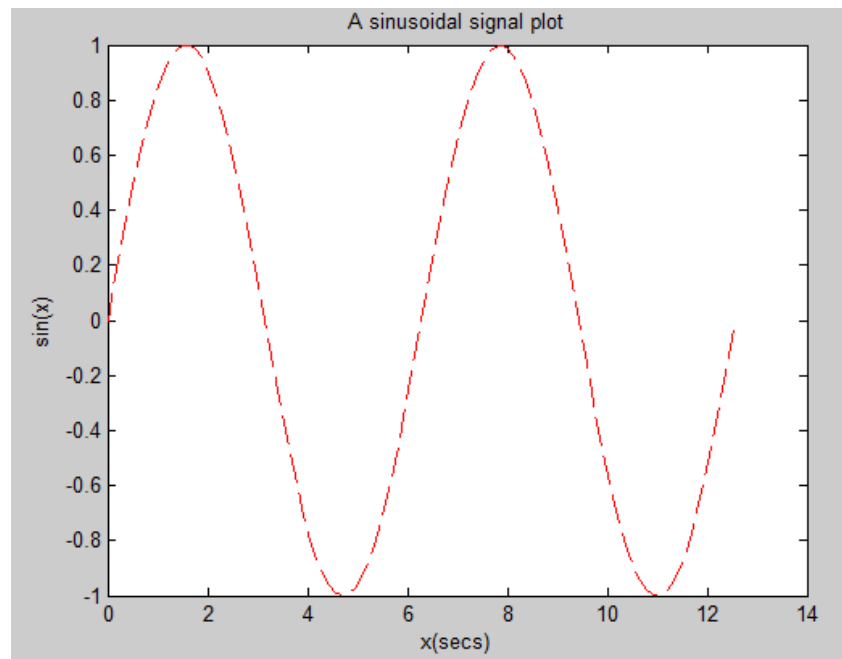
```
y=sin(x);
```

```
figure; plot(x,y,'r--');
```

```
title('A sinusoidal signal plot');
```

```
xlabel('x(secs)');
```

```
ylabel('sin(x)');
```



MATLAB

- subplot komutu ile birden fazla çizim tek pencerede parçalı olarak gösterilebilir

- Örnek:

```
t = 0:pi/10:2*pi;
```

```
[X,Y,Z] = cylinder(4*cos(t));
```

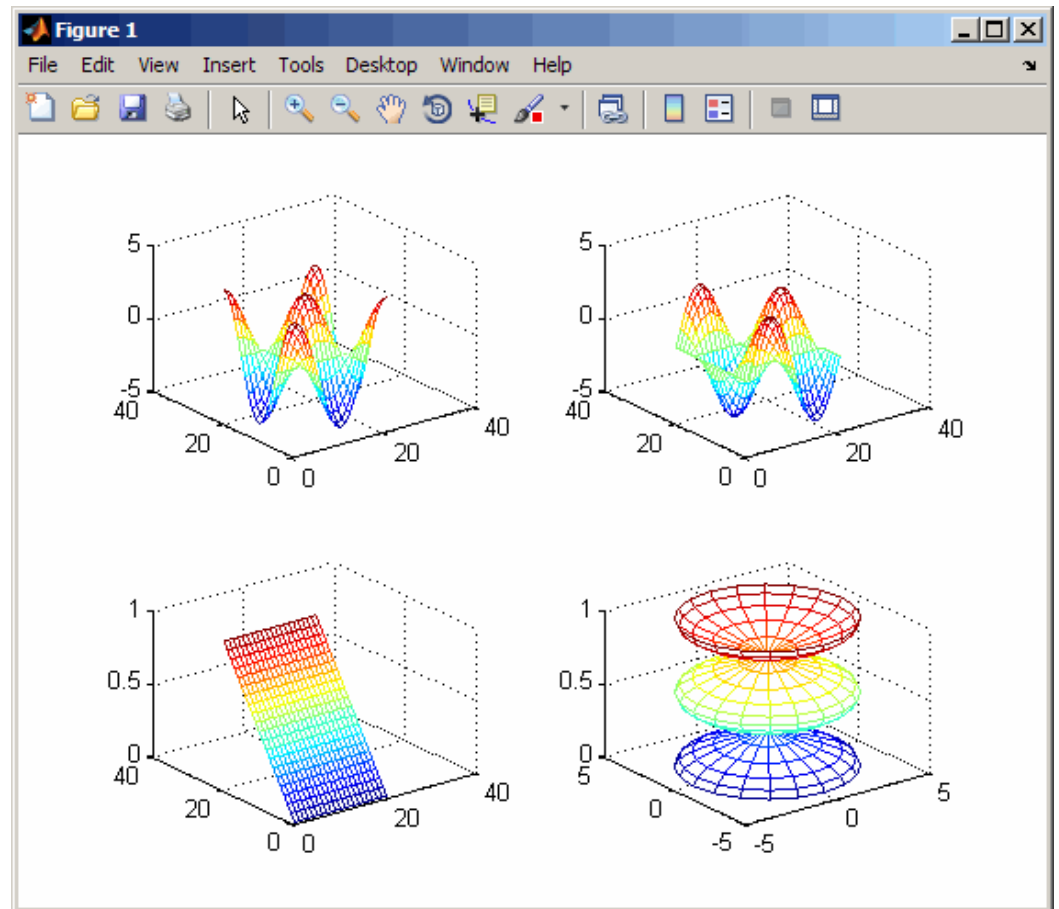
```
figure;
```

```
subplot(2,2,1); mesh(X)
```

```
subplot(2,2,2); mesh(Y)
```

```
subplot(2,2,3); mesh(Z)
```

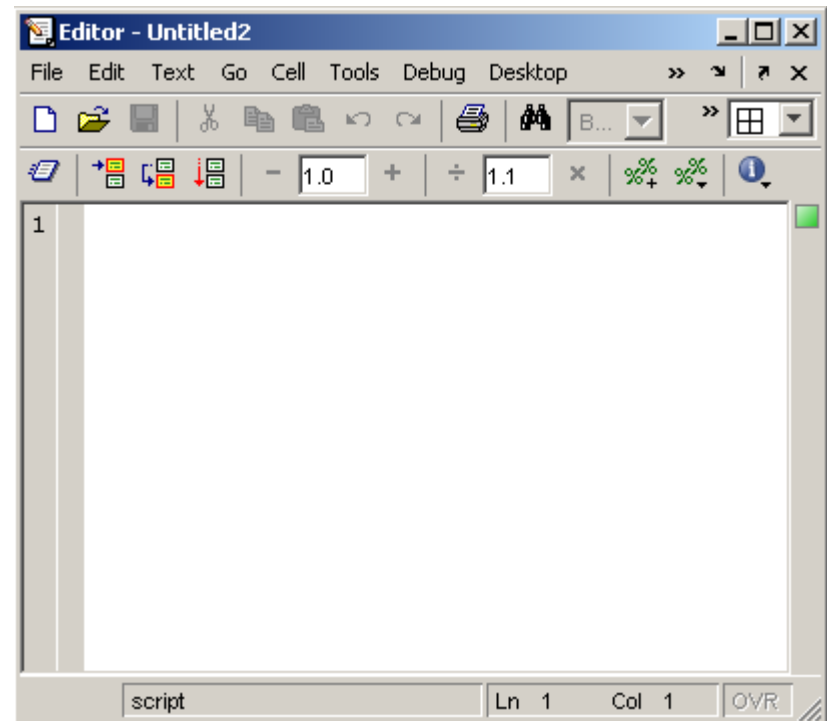
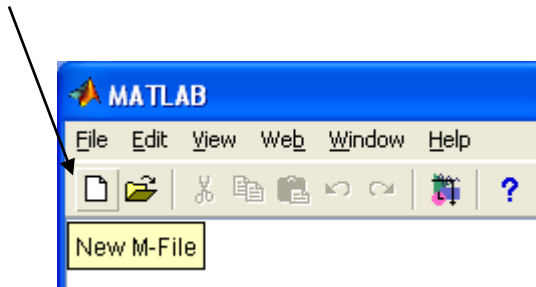
```
subplot(2,2,4); mesh(X,Y,Z)
```



MATLAB

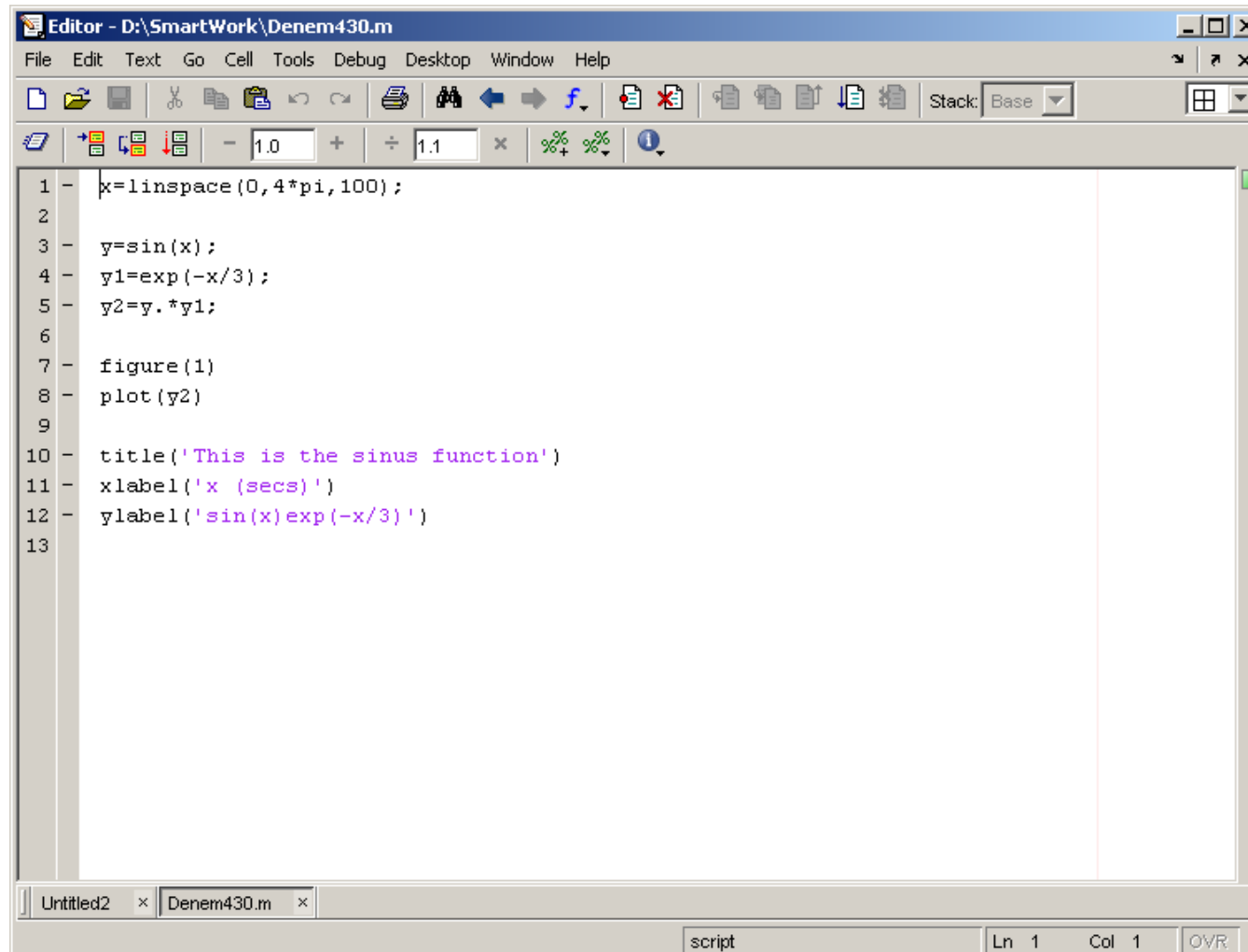
- Fonksiyon yazımı:

Yeni bir .m
uzantılı dosya
yaratmak için



MATLAB

- Fonksiyon yazımı:



The image shows a MATLAB Editor window titled "Editor - D:\SmartWork\Denem430.m". The window contains a script with the following code:

```
1 - x=linspace(0,4*pi,100);  
2  
3 - y=sin(x);  
4 - y1=exp(-x/3);  
5 - y2=y.*y1;  
6  
7 - figure(1)  
8 - plot(y2)  
9  
10 - title('This is the sinus function')  
11 - xlabel('x (secs)')  
12 - ylabel('sin(x)exp(-x/3)')  
13
```

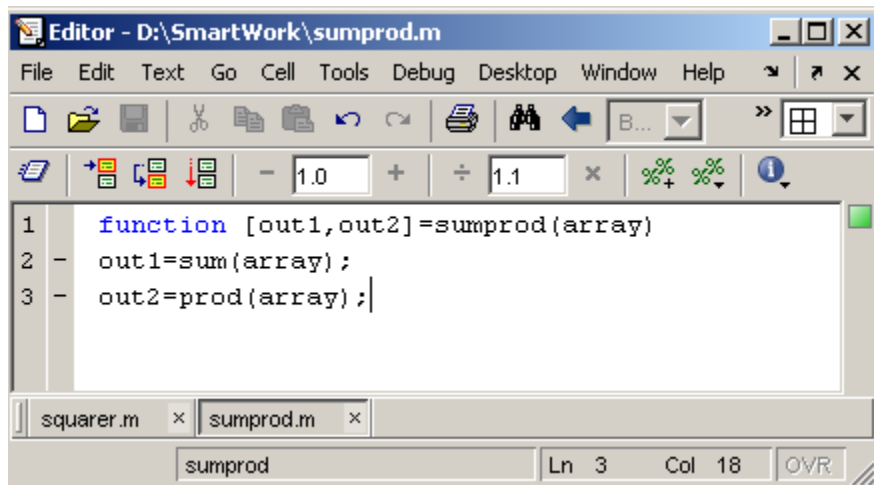
The script defines a vector x from 0 to 4π with 100 points, calculates $y = \sin(x)$ and $y_1 = \exp(-x/3)$, then plots the product $y_2 = y \cdot y_1$. The plot is titled "This is the sinus function" and the axes are labeled "x (secs)" and "sin(x)exp(-x/3)".

MATLAB

- Fonksiyon yazımı:
- `function out1 = functionname(in1)`
- `function out1 = functionname(in1 , in2 , in3)`
- `function [out1 , out2] = functionname(in1 , in2)`

MATLAB

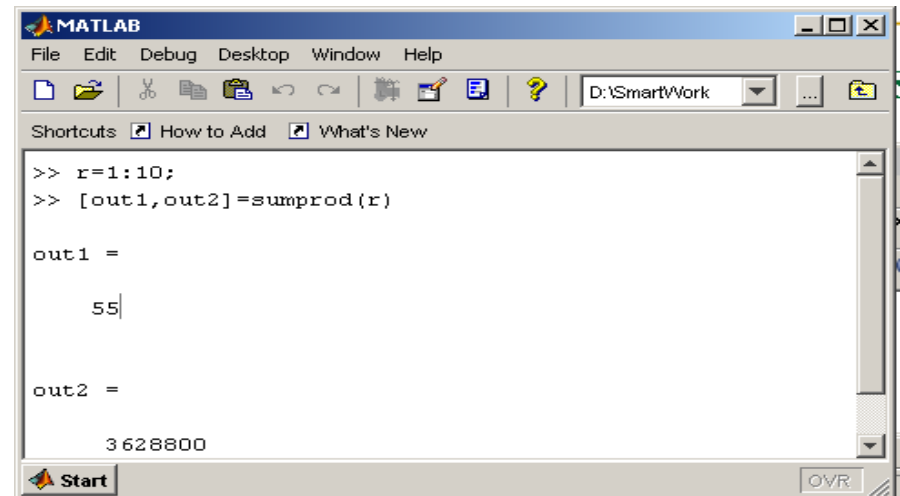
- Girdi olarak bir dizi alan ve çıktı olarak bu dizinin elemanlarının toplamını ve çarpımını veren fonksiyon
- Yazılan sumprod fonksiyonu hem Editor aracılığıyla, hem de komut penceresinden (aynı dizinde yer almak şartıyla) çalıştırılabilir



The image shows the MATLAB Editor window with the file `sumprod.m` open. The code defines a function `sumprod` that takes an array as input and returns its sum and product. The code is as follows:

```
1 function [out1,out2]=sumprod(array)
2 - out1=sum(array);
3 - out2=prod(array);
```

The window title is "Editor - D:\SmartWork\sumprod.m". The status bar at the bottom shows "sumprod", "Ln 3", "Col 18", and "OVR".



The image shows the MATLAB Command Window with the following commands and outputs:

```
>> r=1:10;
>> [out1,out2]=sumprod(r)

out1 =

    55

out2 =

 3628800
```

The window title is "MATLAB". The status bar at the bottom shows "Start" and "OVR".

MATLAB ve İmge İşleme

- Bir görüntü okumak ve göstermek için:

```
I = imread('cameraman.tif');  
figure; imshow(I);
```

- Eğer görüntü mevcut uzantılı klasörde (current directory) değilse (ve MATLAB'ın kendi görüntülerinden biri değilse) dosya uzantısı yazılmalıdır:

```
I = imread('D:\Calismalar\Eski\Image Sequences\Lena.bmp');  
figure; imshow(I);
```

MATLAB ve İmge İşleme

- Görüntü halihazırda renkli ise (3 bantlı ise) `imshow` komutu ile renkli gösterilecektir:

```
I = imread('ngc6543a.jpg');  
figure; imshow(I);
```

- Görüntüleri göstermek için aşağıdaki komutlar da kullanılabilir:
 - `image`
 - `imagesc`

MATLAB ve İmge İşleme

- Görüntülerin 3 renk bileşenleri matrissel yapıda ayrı ayrı da gösterilebilir:

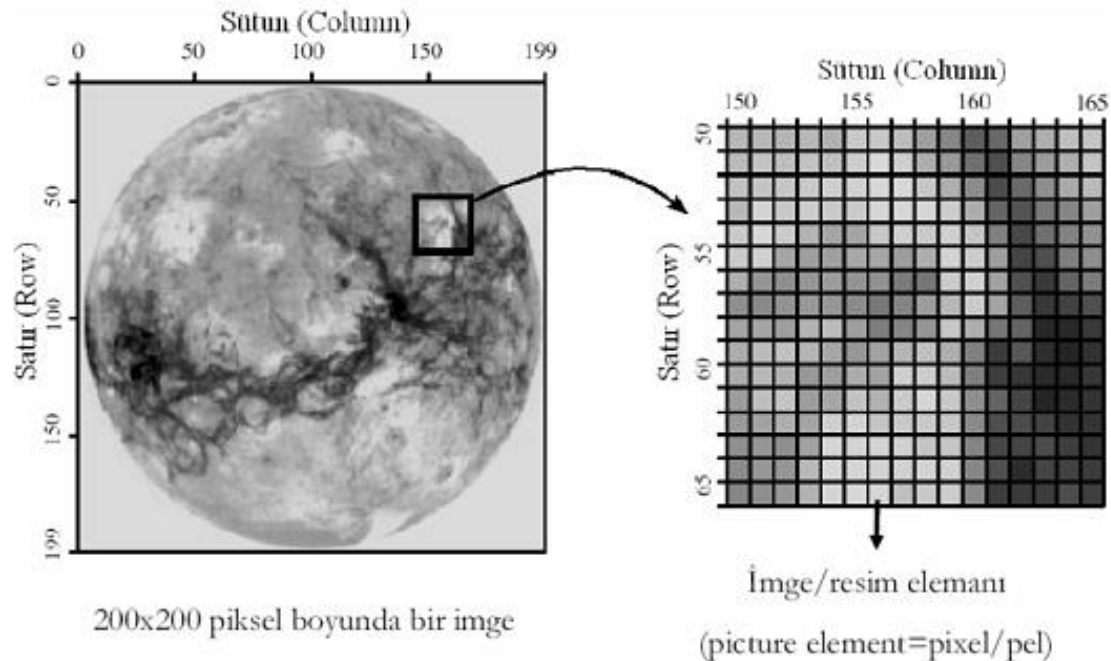
```
I=imread('onion.png');  
figure; imshow(I);  
figure; imshow(I(:,:,1));  
figure; imshow(I(:,:,2));  
figure; imshow(I(:,:,3));
```

```
I = imread('office_5.jpg');  
figure; imshow(I);
```

MATLAB ve İmge İşleme

- Sayısal imge gösterimi:

$$\mathbf{A} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \vdots & \vdots & & \vdots \\ a_{M-1,0} & a_{M-1,1} & \cdots & a_{M-1,N-1} \end{bmatrix}$$



MATLAB ve İmge İşleme

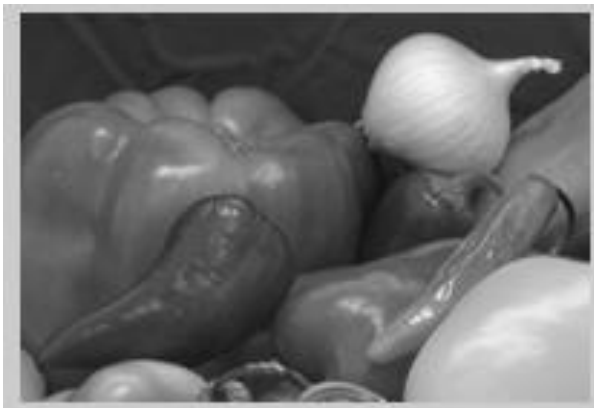
- Devrik İşlemi (*Transpose*):

```
I=imread('onion.png');
```

```
I2 = rgb2gray(I);
```

```
figure; imshow(I2);
```

```
figure; imshow(I2');
```



MATLAB ve İmge İşleme

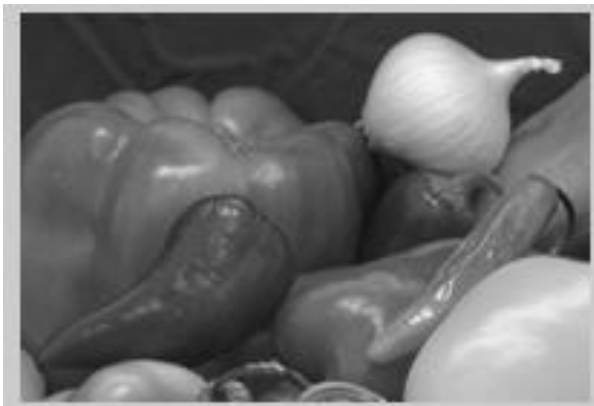
- Yatayda Aynalama:

```
I=imread('onion.png');
```

```
I2 = rgb2gray(I);
```

```
figure; imshow(I2);
```

```
figure; imshow(fliplr(I2));
```



MATLAB ve İmge İşleme

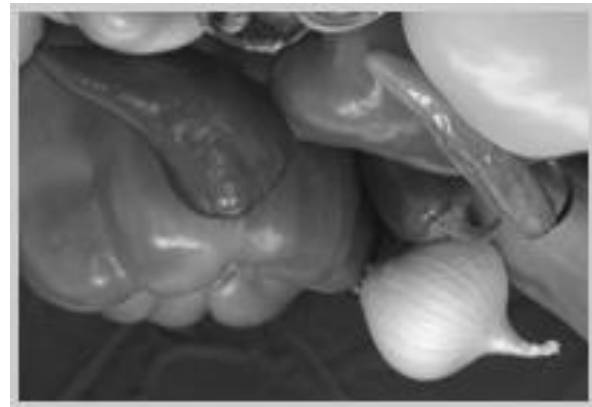
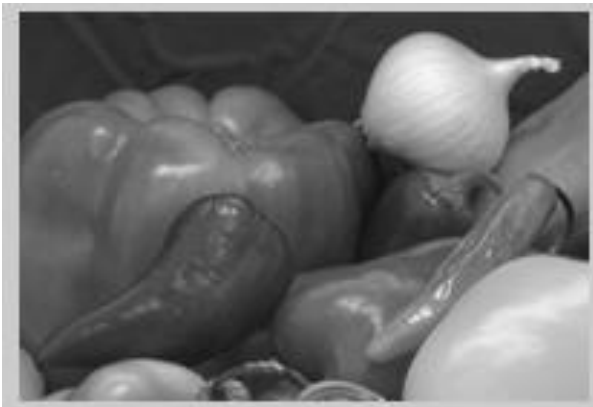
- Düşeyde Aynalama:

```
I=imread('onion.png');
```

```
I2 = rgb2gray(I);
```

```
figure; imshow(I2);
```

```
figure; imshow(flipud(I2));
```



MATLAB ve İmge İşleme

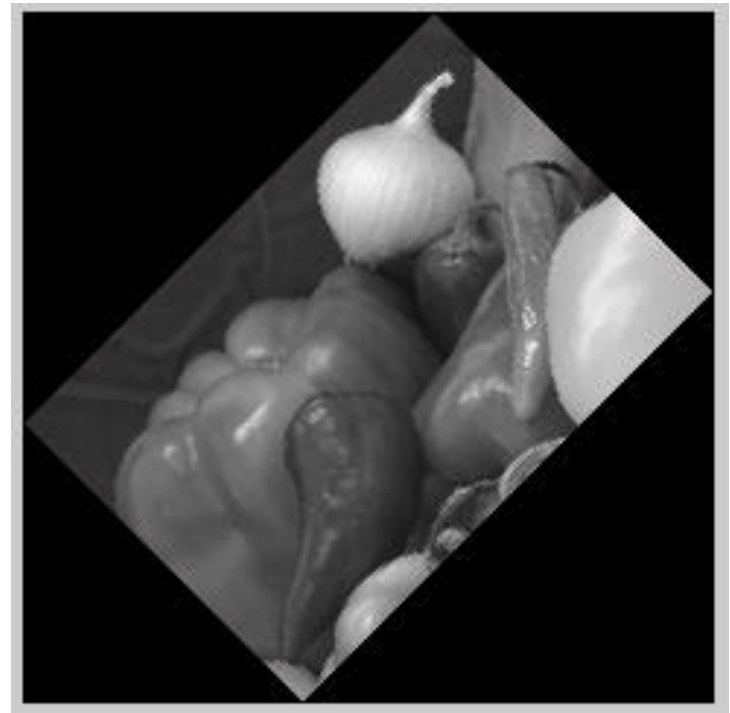
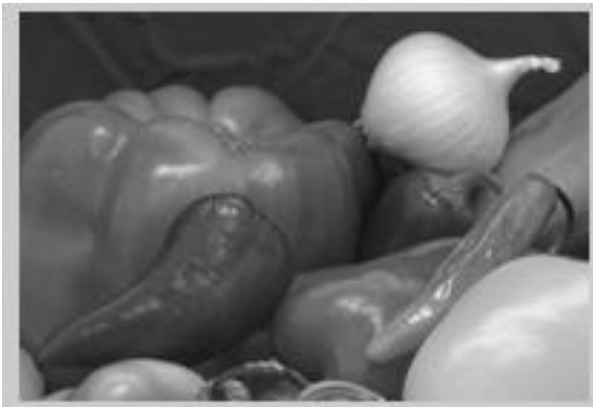
- Döndürme:

```
I=imread('onion.png');
```

```
I2 = rgb2gray(I);
```

```
figure; imshow(I2);
```

```
figure; imshow(imrotate(I2,45));
```



MATLAB ve İmge İşleme

- Döndürme:

```
figure;
```

```
subplot(1,3,1);imshow(imrotate(I2,45,'nearest'));
```

```
subplot(1,3,2);imshow(imrotate(I2,45,'bilinear'));
```

```
subplot(1,3,3);imshow(imrotate(I2,45,'bicubic'));
```

```
figure;
```

```
subplot(1,3,1);imshow(imrotate(I2,45,'bicubic'));
```

```
subplot(1,3,2);imshow(imrotate(I2,45,'bicubic','loose'));
```

```
subplot(1,3,3);imshow(imrotate(I2,45,'bicubic','crop'));
```

MATLAB ve İmge İşleme

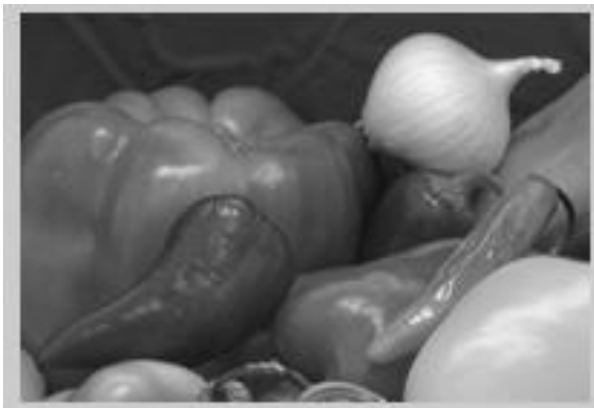
- Kesme:

```
I=imread('onion.png');
```

```
I2 = rgb2gray(I);
```

```
figure; imshow(I2);
```

```
figure; imshow(I2(6:60,110:190));
```



MATLAB ve İmge İşleme

- Işıklılık Değişimi:

```
I=imread('cameraman.tif');  
figure; subplot(1,3,1); imshow(I);  
subplot(1,3,2); imshow(I+60);  
subplot(1,3,3); imshow(I-60);
```



MATLAB ve İmge İşleme

- Negatif alma:

```
I=imread('cameraman.tif');  
figure; subplot(1,2,1); imshow(I);  
subplot(1,2,2); imshow(255-I);
```



MATLAB ve İmge İşleme

- Zıtlık (*Contrast*) Değişimi:

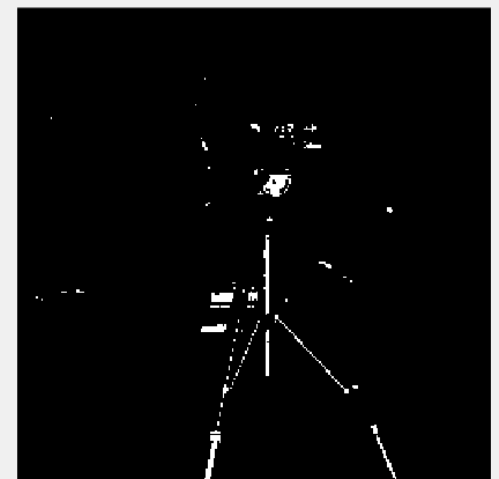
```
I=imread('cameraman.tif');  
figure; subplot(1,3,1); imshow(I);  
subplot(1,3,2); imshow(I*1.5);  
subplot(1,3,3); imshow(I*0.5);
```



MATLAB ve İmge İşleme

- Eşikleme:

```
I=imread('cameraman.tif');  
figure; subplot(1,3,1); imshow(I);  
subplot(1,3,2); imshow(I>100);  
subplot(1,3,3); imshow(I>200);
```



MATLAB ve İmge İşleme

- Histogram:
- Her ışıklılık seviyesinin görüntüde bulunma sayısını (piksel sayısı) veya oranını, konum bilgisinden bağımsız olarak, verir

```
I=imread('cameraman.tif');  
figure; imhist(I);
```


MATLAB ve İmge İşleme

- Histogram:

```
I=imread('cameraman.tif');
```

```
figure; imhist(I);
```

```
I2 = I+60;
```

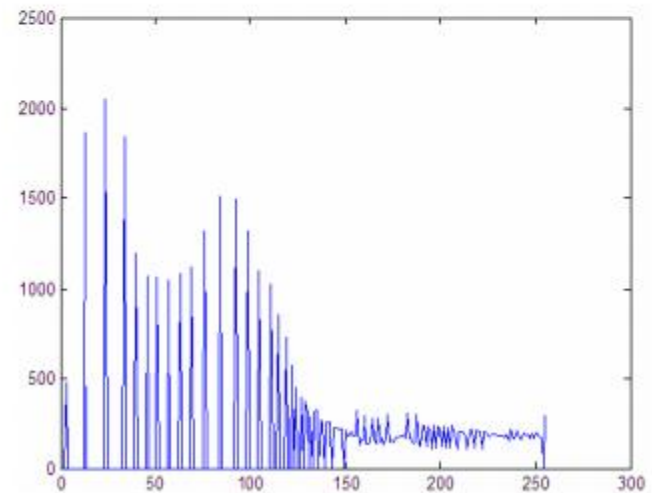
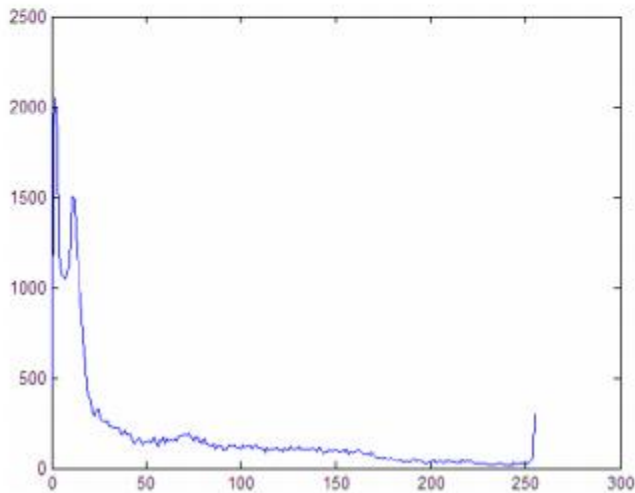
```
figure; imhist(I2);
```

```
I3 = I *0.5;
```

```
figure; imhist(I3);
```

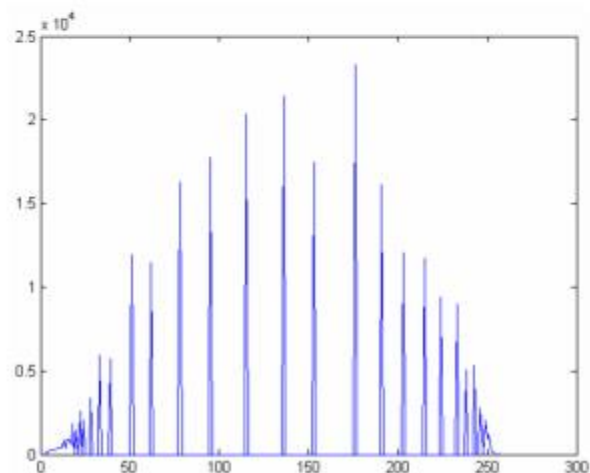
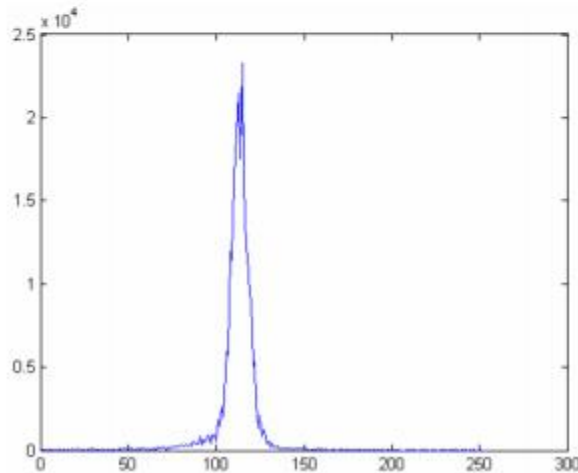
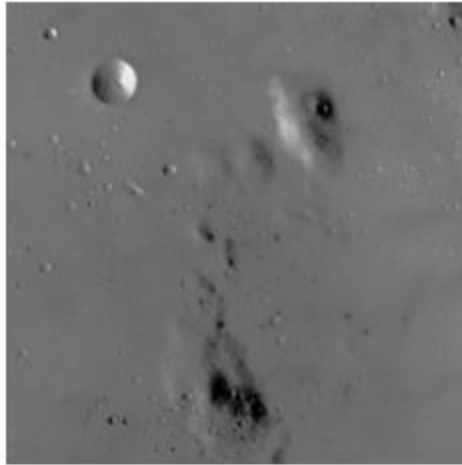
MATLAB ve İmge İşleme

- Histogram Eşitleme (*Equalization*):



MATLAB ve İmge İşleme

- Histogram Eşitleme (*Equalization*):



MATLAB ve İmge İşleme

- Histogram Eşitleme (*Equalization*):

```
I = imread('D:\Calismalar\Eski\Image Sequences\rot1.bmp');  
figure; subplot(1,2,1); imshow(I);  
subplot(1,2,2); imshow(histeq(I));
```

- Not: Görüntü MATLAB'ın içkin görüntülerinden biri değil

MATLAB ve İmge İşleme

- Histogram Eşitleme (*Equalization*):

```
I = imread('D:\Calismalar\Eski\Image Sequences\rot1.bmp');  
figure; subplot(1,2,1); imshow(I);  
subplot(1,2,2); imshow(histeq(I));
```



MATLAB ve İmge İşleme

- Histograma göre Eşik Seçimi:

```
clear all; close all; clc;
I = imread('cameraman.tif');
[H,W] = size(I);
figure, imshow(I);

[hist, x] = imhist(I);
figure, imhist(I);
p = hist / (H*W);
t = 128;
for iter = 1:1:100
    disp(['Threshold is: ' num2str(t)]);
    m1 = (x > t);
    u1 = sum( x(m1) .* p(m1) ) / sum(p(m1));
    m2 = (x <= t);
    u2 = sum( x(m2) .* p(m2) ) / sum(p(m2));
    tnew = (u1 + u2)/2;
    if t == tnew
        break;
    else
        t = tnew;
    end
    figure; imshow(I>t);
end
figure; subplot(1,2,1); imshow(I);
subplot(1,2,2); imshow(I > t);
```

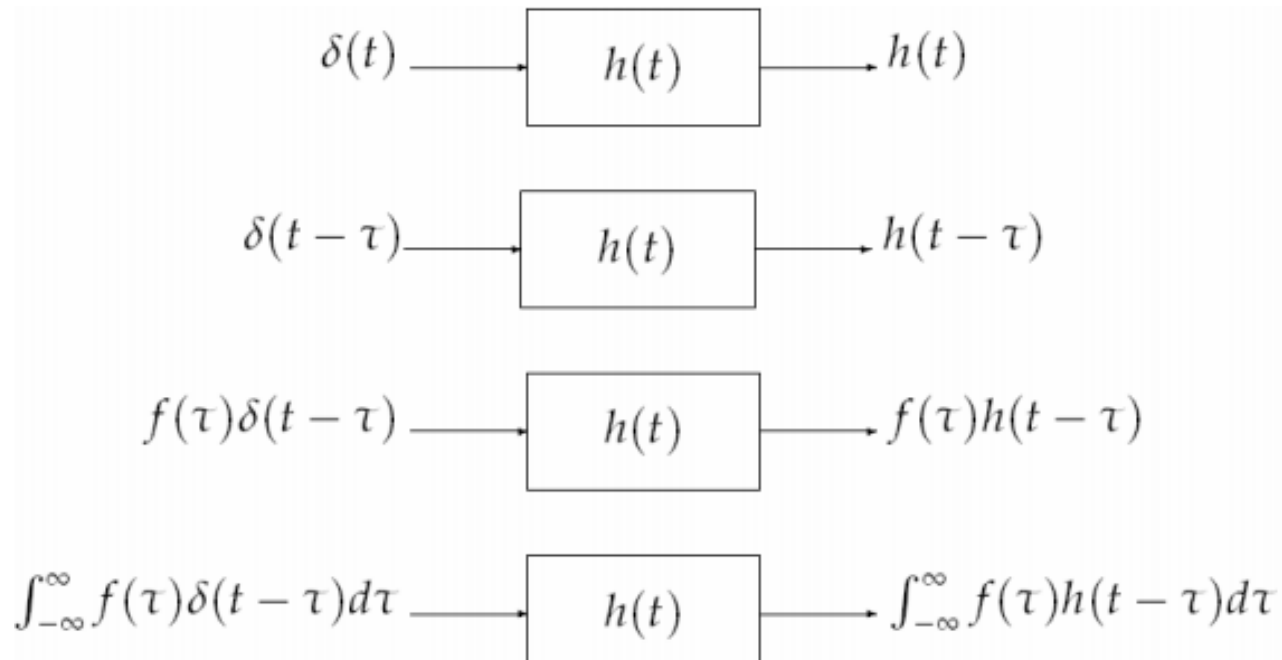
MATLAB ve İmge İşleme

- Histograma göre Eşik Seçimi:



MATLAB ve İmge İşleme

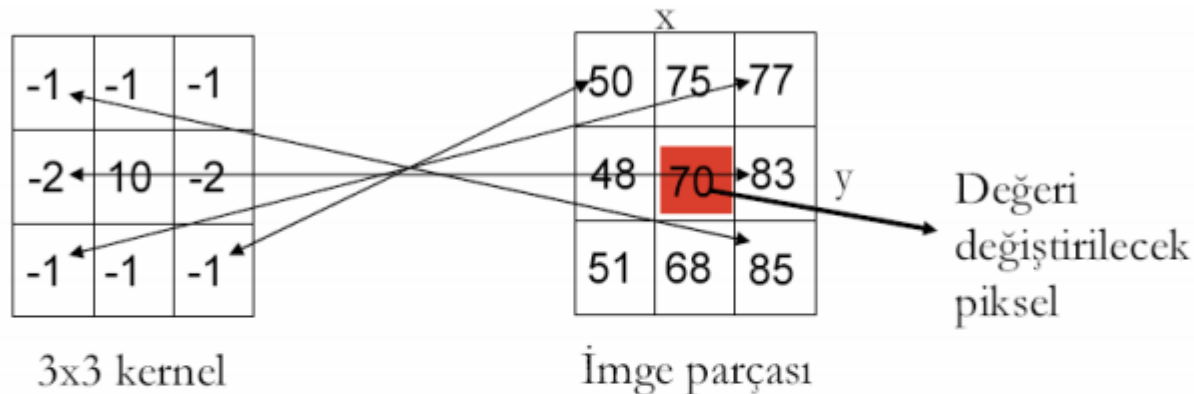
- Evrişim (*Convolution*):



MATLAB ve İmge İşleme

- Evrişim (*Convolution*):

$$g(x,y) = k * f = \sum_{i=-m}^m \sum_{j=-n}^n k(i,j) f(x-i, y-j)$$



$$\begin{aligned} g(x,y) = & k(-1,-1)f(x+1,y+1) + k(0,-1)f(x,y+1) + k(1,-1)f(x-1,y+1) + k(-1,0)f(x+1,y) \\ & + k(0,0)f(x,y) + k(1,0)f(x-1,y) + k(-1,1)f(x+1,y-1) + k(0,1)f(x,y-1) \\ & + k(1,1)f(x-1,y-1) \end{aligned}$$

$$\begin{aligned} g(x,y) = & (-1 \times 85) + (-1 \times 68) + (-1 \times 51) + (-2 \times 83) + (10 \times 70) + (-2 \times 48) + (-1 \times 77) \\ & + (-1 \times 75) + (-1 \times 50) = 32 \end{aligned}$$

MATLAB ve İmge İşleme

- Evrişim (*Convolution*):

```
I=imread('cameraman.tif');
```

```
I2 = imfilter(I,[1 1 1; 1 1 1; 1 1 1]/9);
```

```
figure; subplot(1,2,1); imshow(I);
```

```
subplot(1,2,2); imshow(I2);
```



MATLAB ve İmge İşleme

- Evrişim (*Convolution*):

```
I=imread('cameraman.tif');
```

```
I2 = imfilter(I,[-1 -1 -1; 0 0 0; 1 1 1]);
```

```
figure; subplot(1,2,1); imshow(I);
```

```
subplot(1,2,2); imshow(I2);
```



MATLAB ve İmge İşleme

- Evrişim (*Convolution*):

```
I=imread('cameraman.tif');
```

```
I2 = imfilter(I,[-1 0 1; -1 0 1; -1 0 1]);
```

```
figure; subplot(1,2,1); imshow(I);
```

```
subplot(1,2,2); imshow(I2);
```



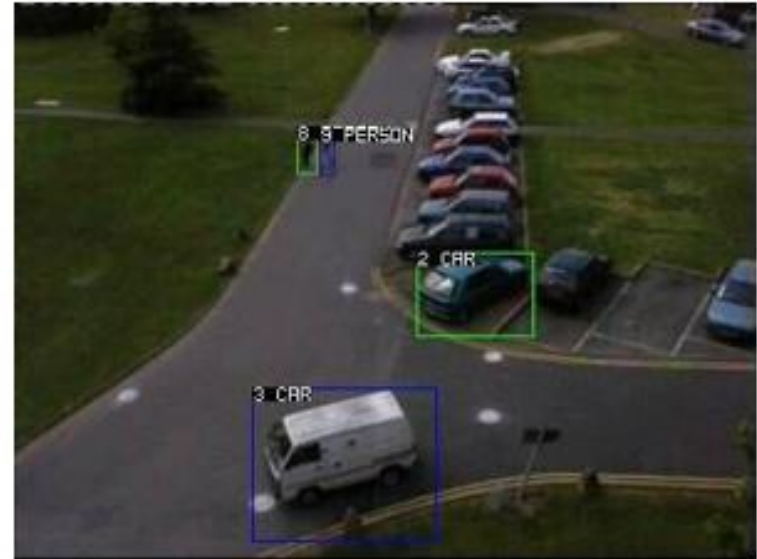
İmge İşleme: Tespit



İmge İşleme: Tespit



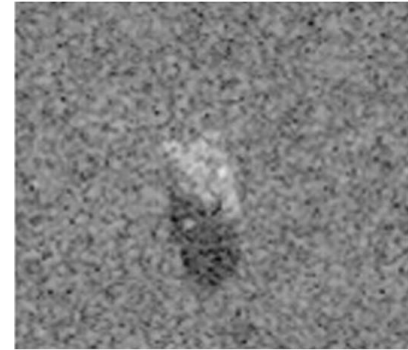
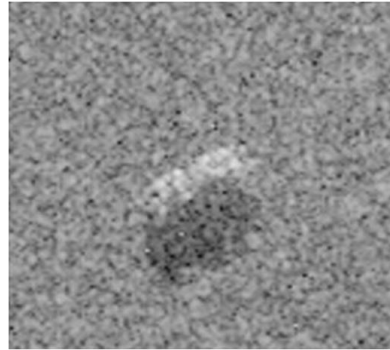
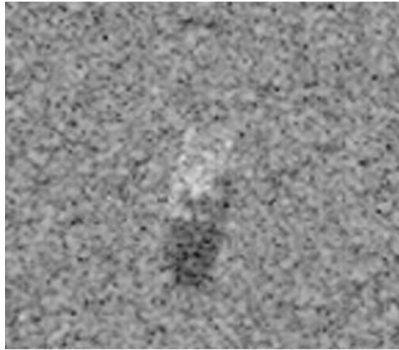
İmge İşleme: Tespit



İmge İşleme: Tanıma



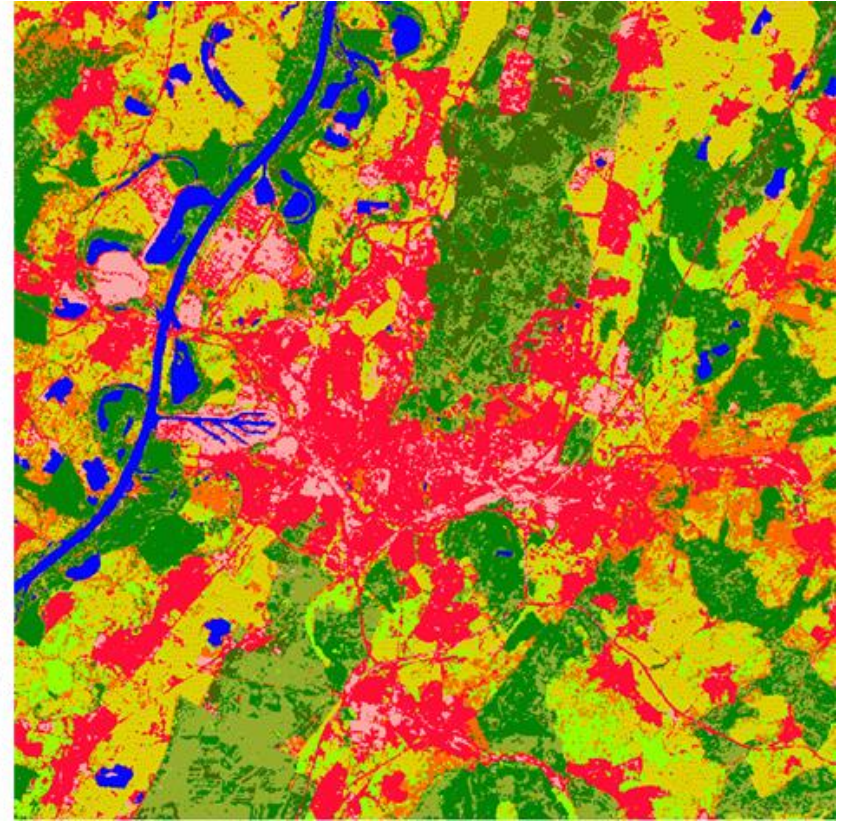
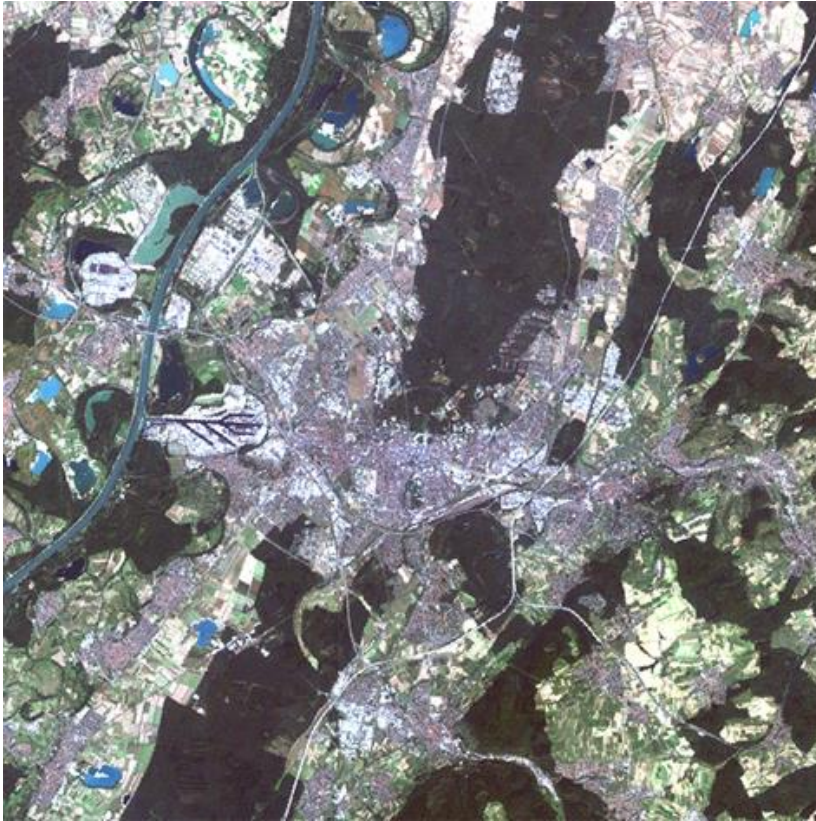
İmge İşleme: Tanıma



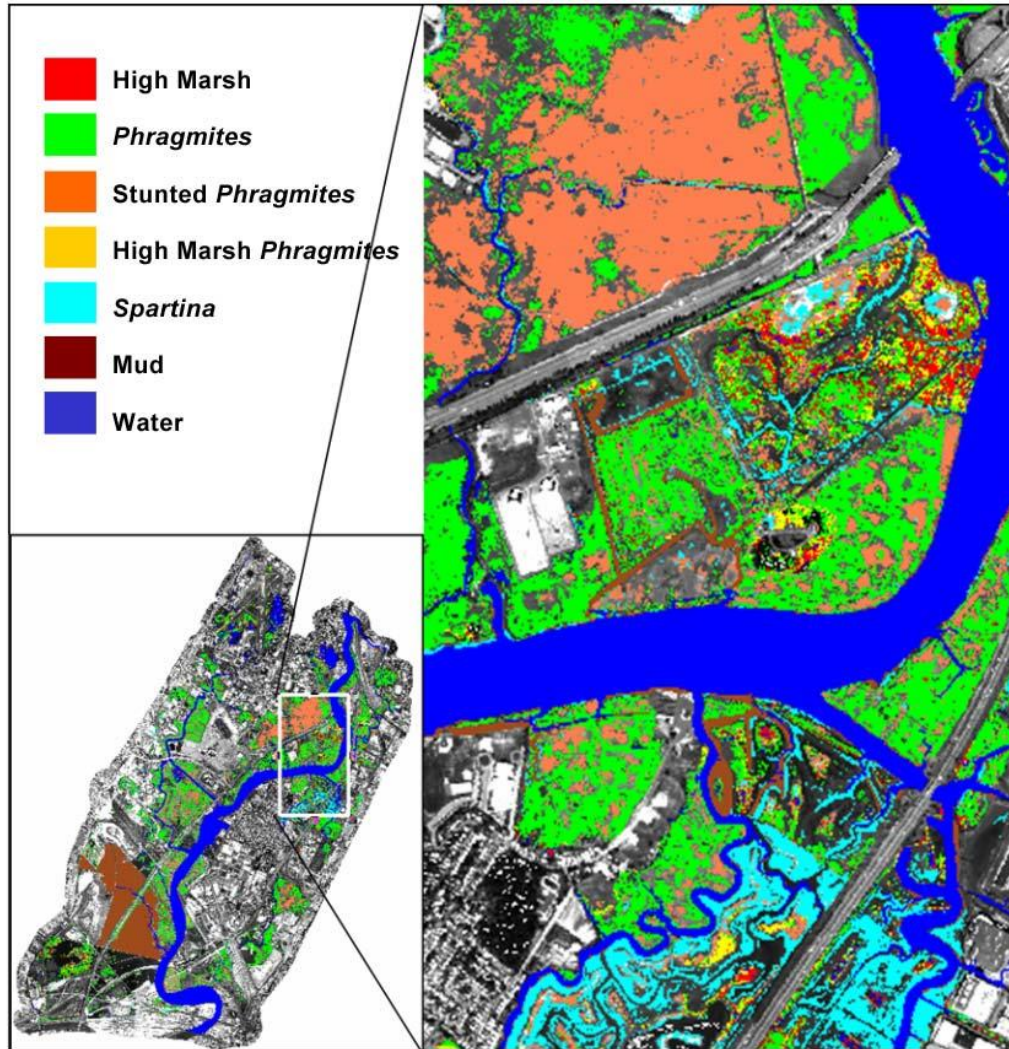
İmge İşleme: Sınıflandırma



İmge İşleme: Sınıflandırma



İmge İşleme: Sınıflandırma



Visual recognition

Verification

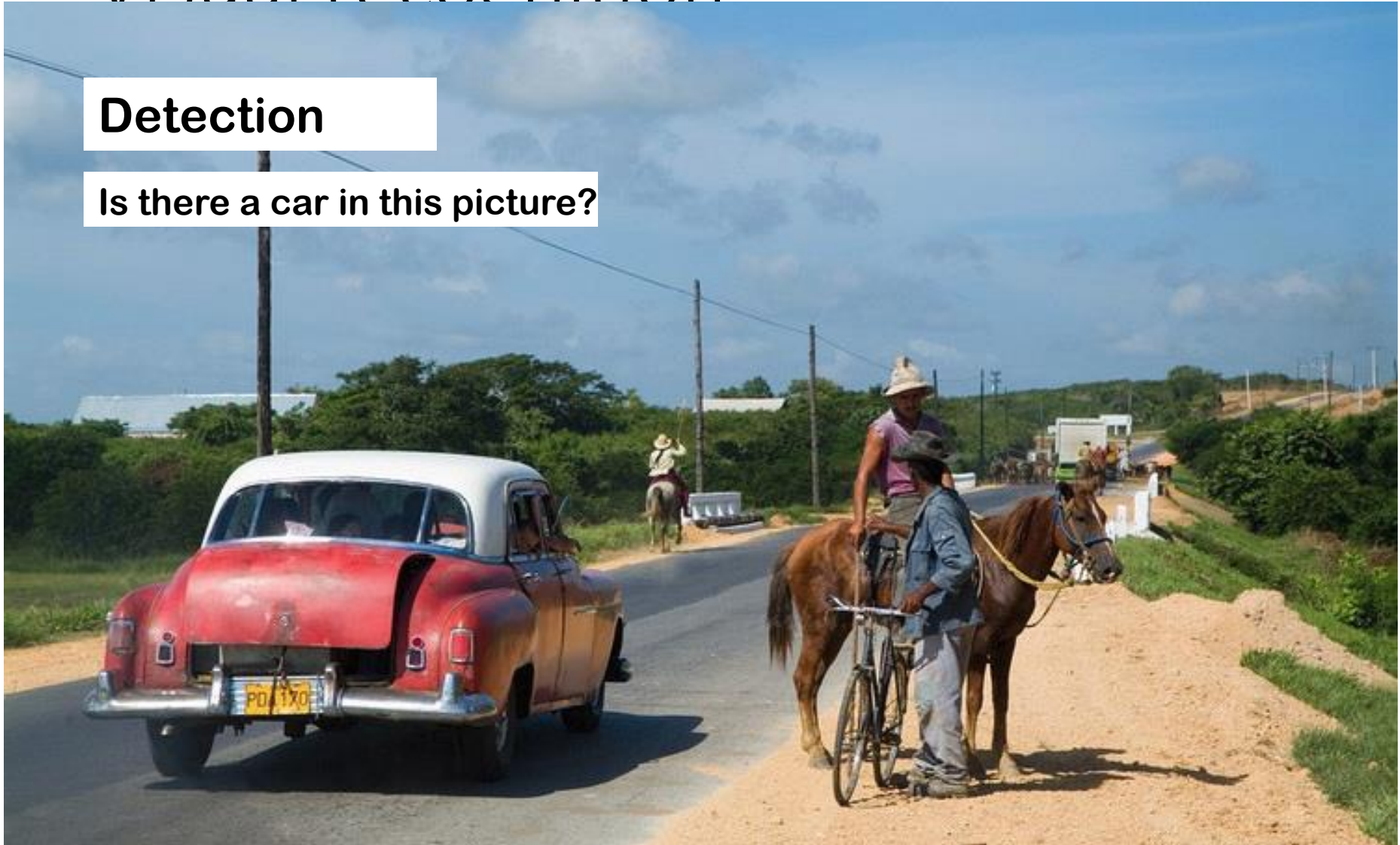
Is this a car?



Visual recognition

Detection

Is there a car in this picture?



Visual recognition

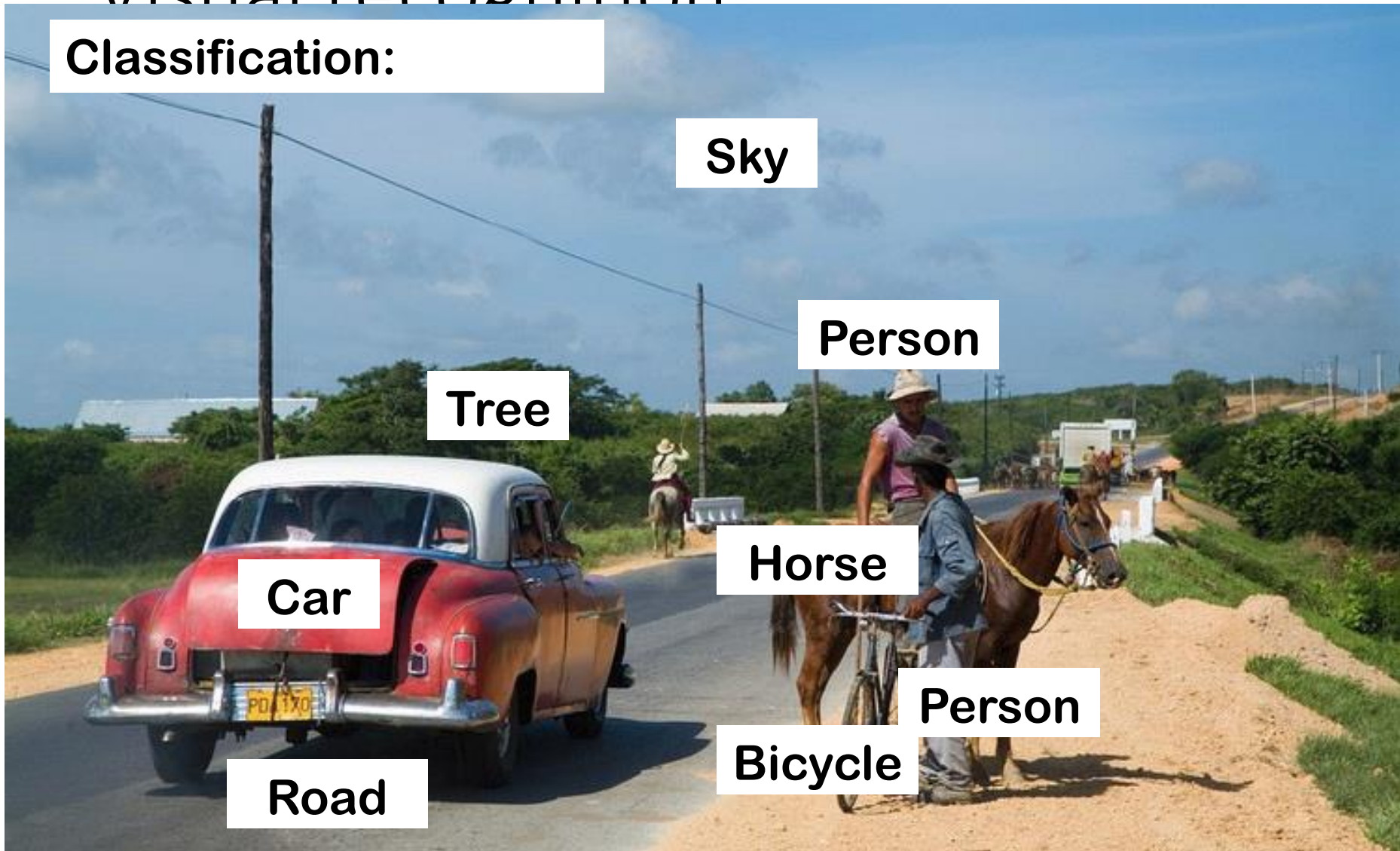
Detection

Where is the car in this picture?



Visual recognition

Classification:



Sky

Person

Tree

Car

Horse

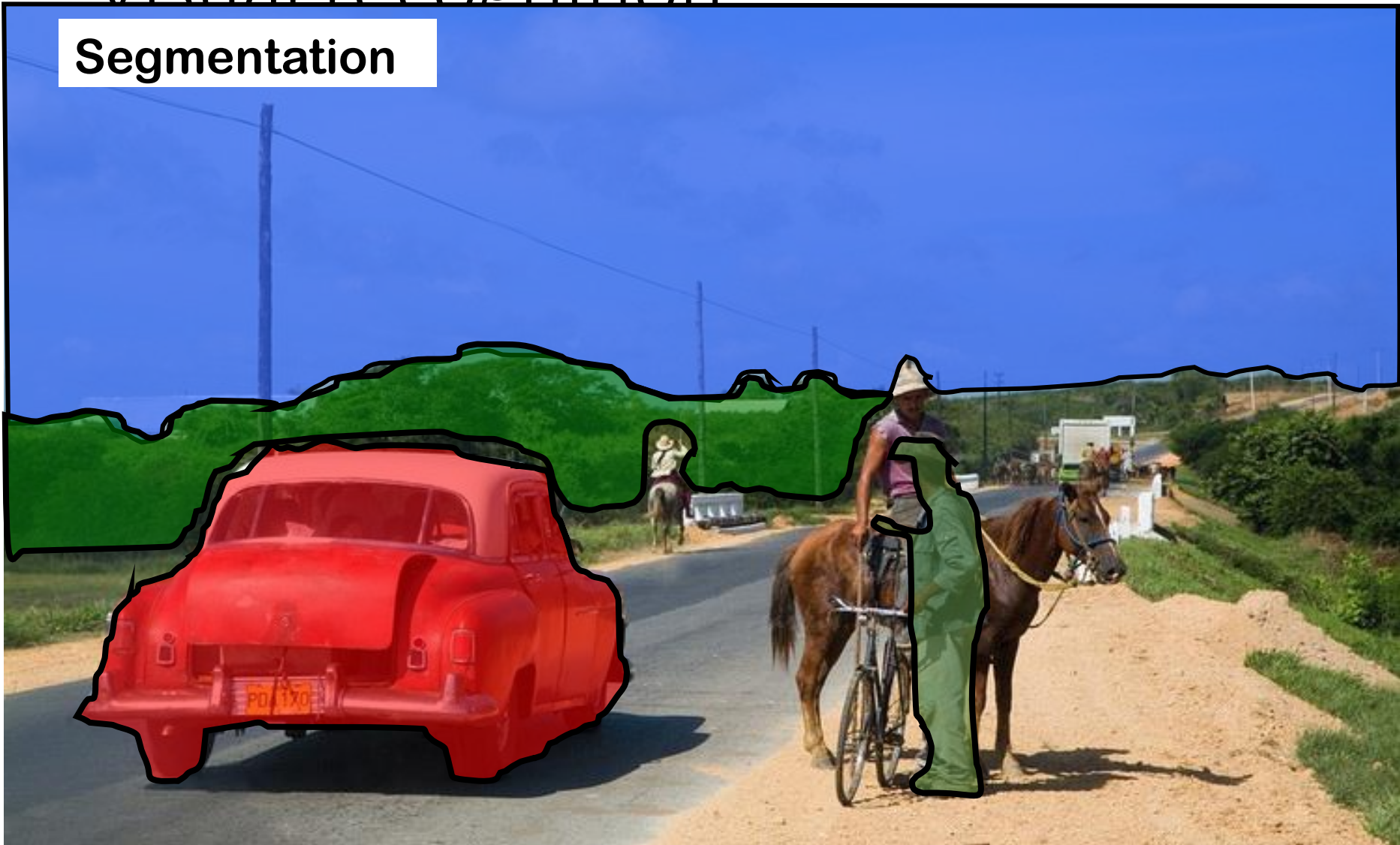
Person

Bicycle

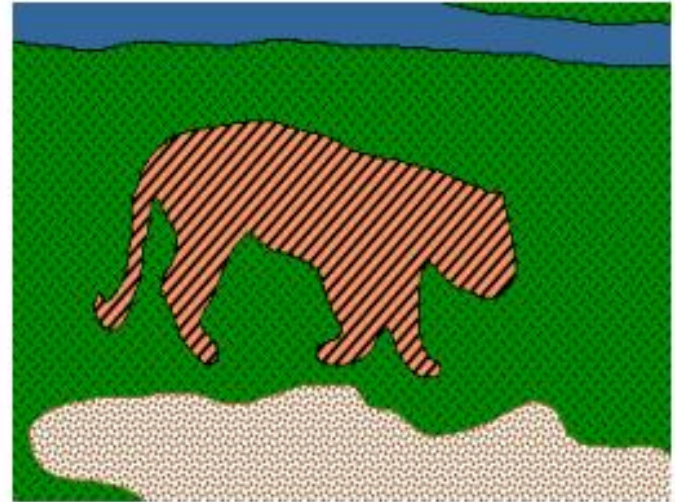
Road

Visual recognition

Segmentation



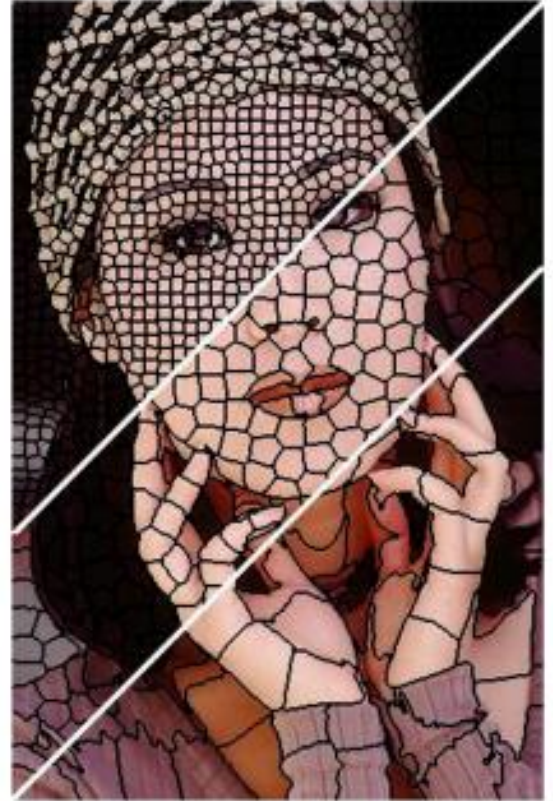
İmge İşleme: Kümeleme



İmge İşleme: Kümeleme



İmge İşleme: Kümeleme



İmge İşleme: Kümeleme

- Örnek yöntem: k - Ortalamalar Kümeleme
- Algoritma:
 - Uzayda dağılmış veri üzerinden K adet nokta belirlenir ve bu noktalar ilk küme merkezleri olarak alınır
 - Her veri noktası, kendisine değer olarak en yakın merkezli kümeye atanır
 - Küme merkezleri kendilerine atanan veri noktalarının ortalamaları olarak güncellenir
 - İşlem, küme merkezleri değişmez hale gelene kadar, veya yineleme sayısı maksimum sayıya ulaşana kadar devam eder

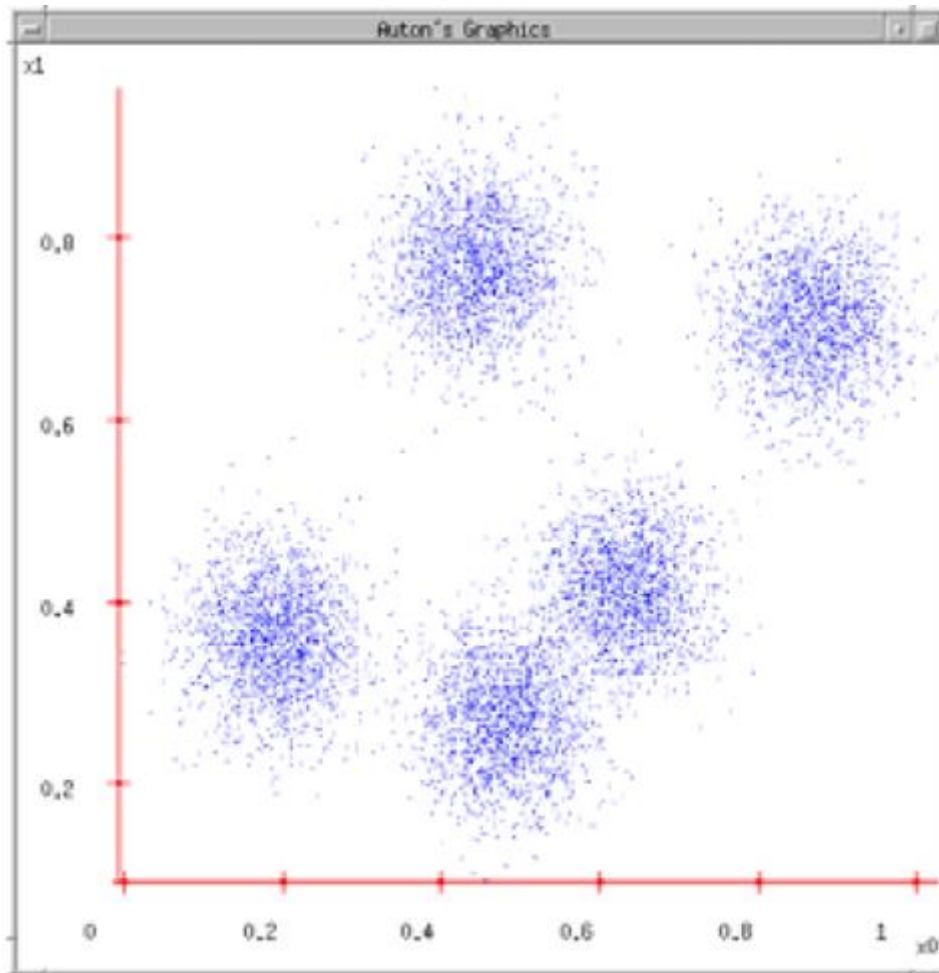
İmge İşleme: Kümeleme

- Örnek yöntem: k - Ortalamalar Kümeleme



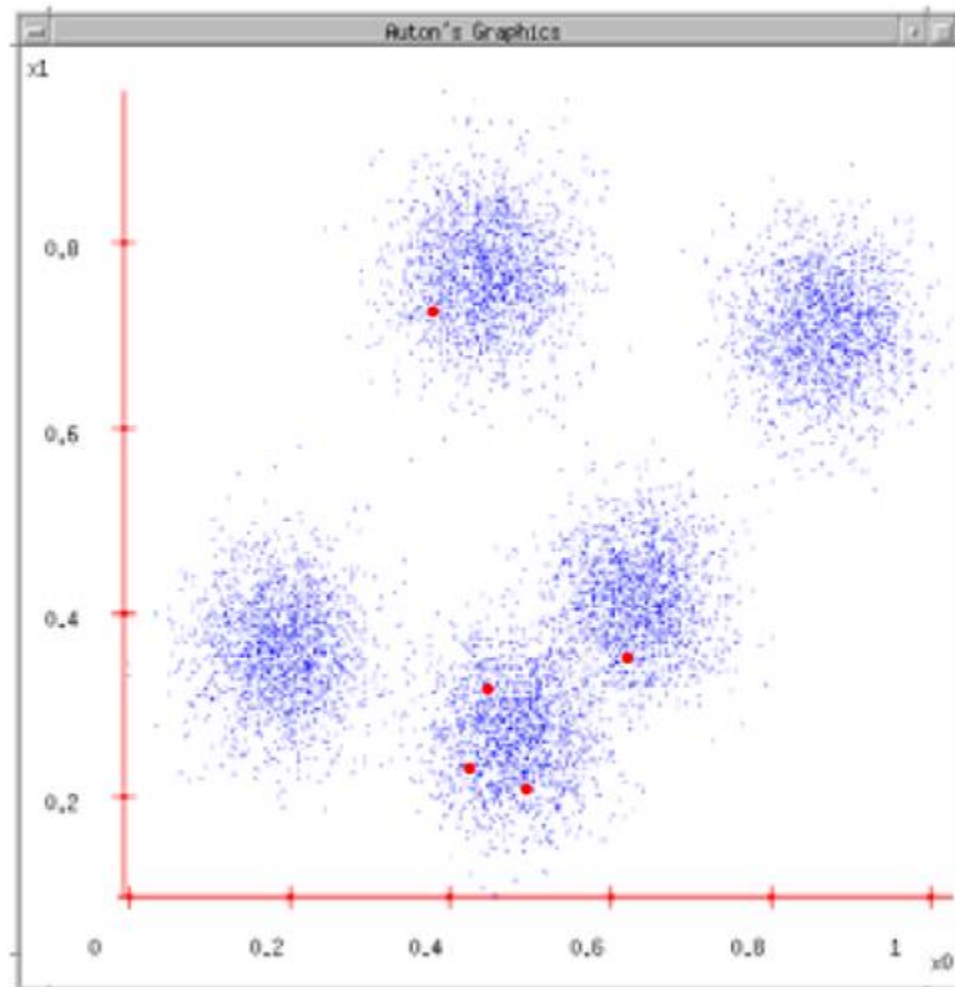
İmge İşleme: Kümeleme

- Örnek yöntem: k - Ortalamalar Kümeleme



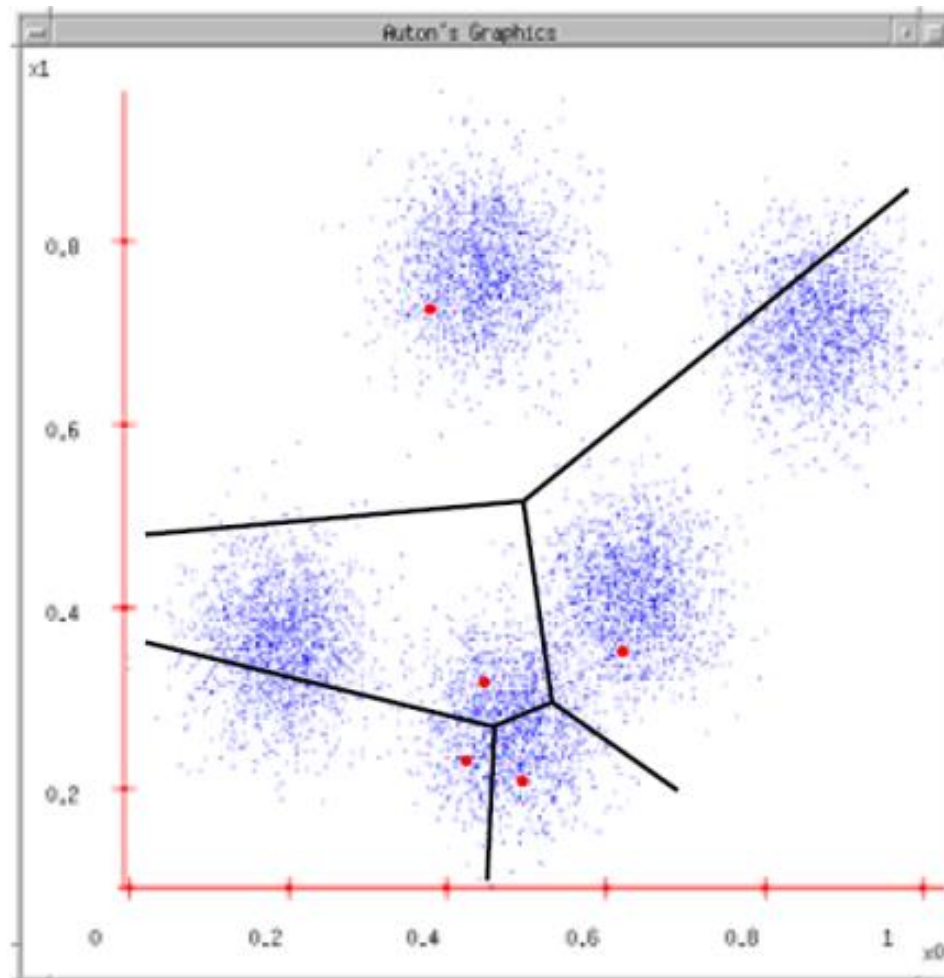
İmge İşleme: Kümeleme

- Örnek yöntem: k - Ortalamalar Kümeleme



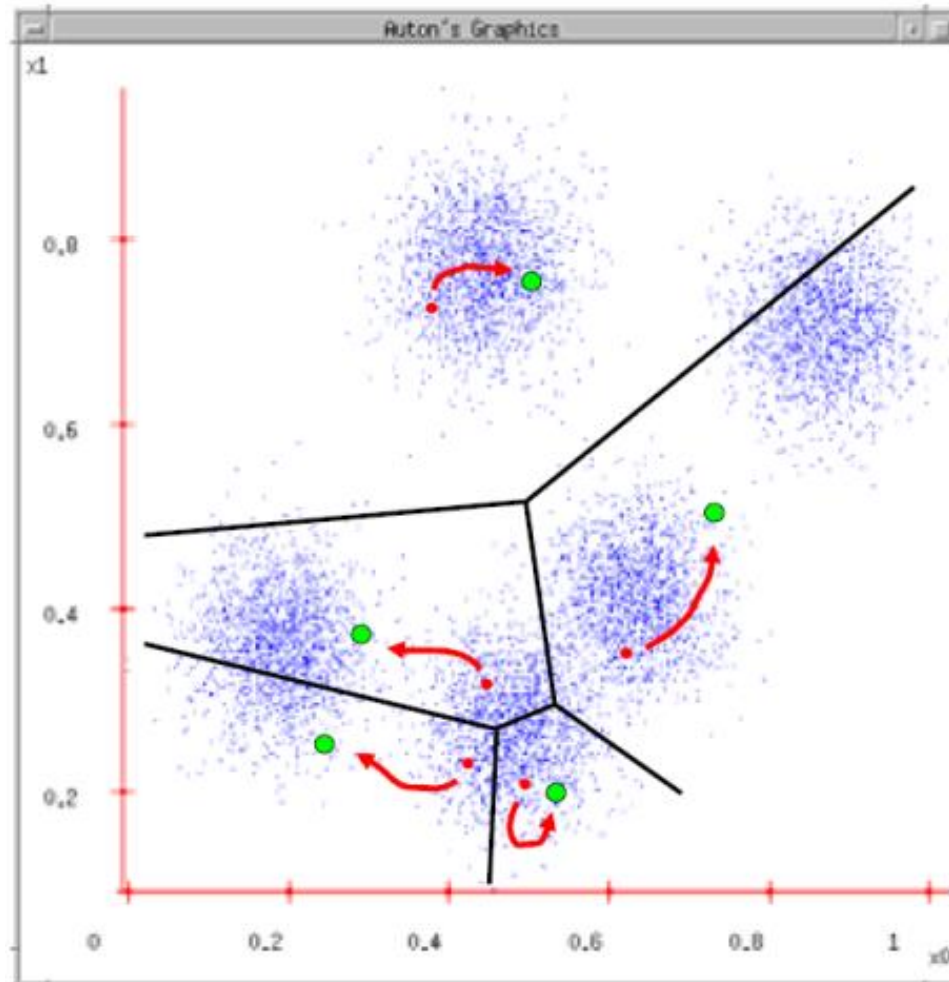
İmge İşleme: Kümeleme

- Örnek yöntem: k - Ortalamalar Kümeleme



İmge İşleme: Kümeleme

- Örnek yöntem: k - Ortalamalar Kümeleme



İmge İşleme: Kümeleme

- Örnek yöntem: k - Ortalamalar Kümeleme



Sorular

?

?

?

?

?