
وتعرف الحكومة الالكترونية بأنها

)

.(

(MATLAB)

Modeling software system to verify identity through iris

Abstract

The revolutionary technology of digital communications delivered more pressure on many public sector institutions to convert their operations to the world of e-business is known as electronic government, known as e-government as including the integrative effective use for all information and communication technologies to facilitate the administrative processes of daily government sectors. The vital features that can be able to experience the people and between those vital features

* مدرس/ قسم بحوث العمليات والتقنيات الذكائية / كلية علوم الحاسوب والرياضيات / جامعة الموصل.

تاريخ القبول 2012/4/23

تاريخ التسليم 2011/11/24

(fingerprints, face, hand geometry, iris scan). This research aims to use iris scans to identify people through the image of the

.....

iris and the application program using the package (MATLAB) to match the images iris for people as it has matching images with the images stored in the database if the amount of the difference between the two images was equal to zero the number of the compared images that were compared It is compared to all the images were obtained a high percentage of the corresponding images.

. (2007)

.(2007)

):

.(-)

.(2005)

1987

1949

Daugman

Daugman

30

1994

Daugman

1995

50

(Daugman, 2007).

1997

(Masek, 2003)

.....

(2002 Daouk et al.)

60

.%96

-: _____

. (Christel, 2010)

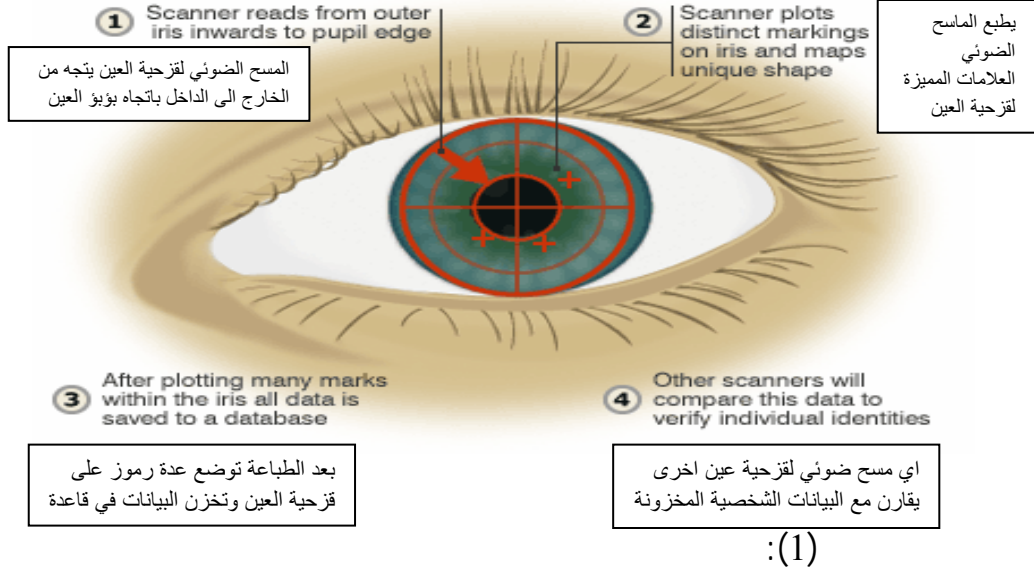
:

-: _____ -

(1982)

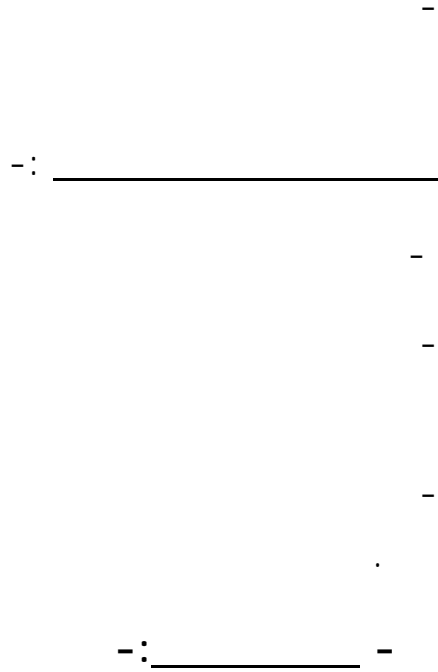
.(1)

HOW IRIS SCANNERS RECORD IDENTITIES



-(2009)

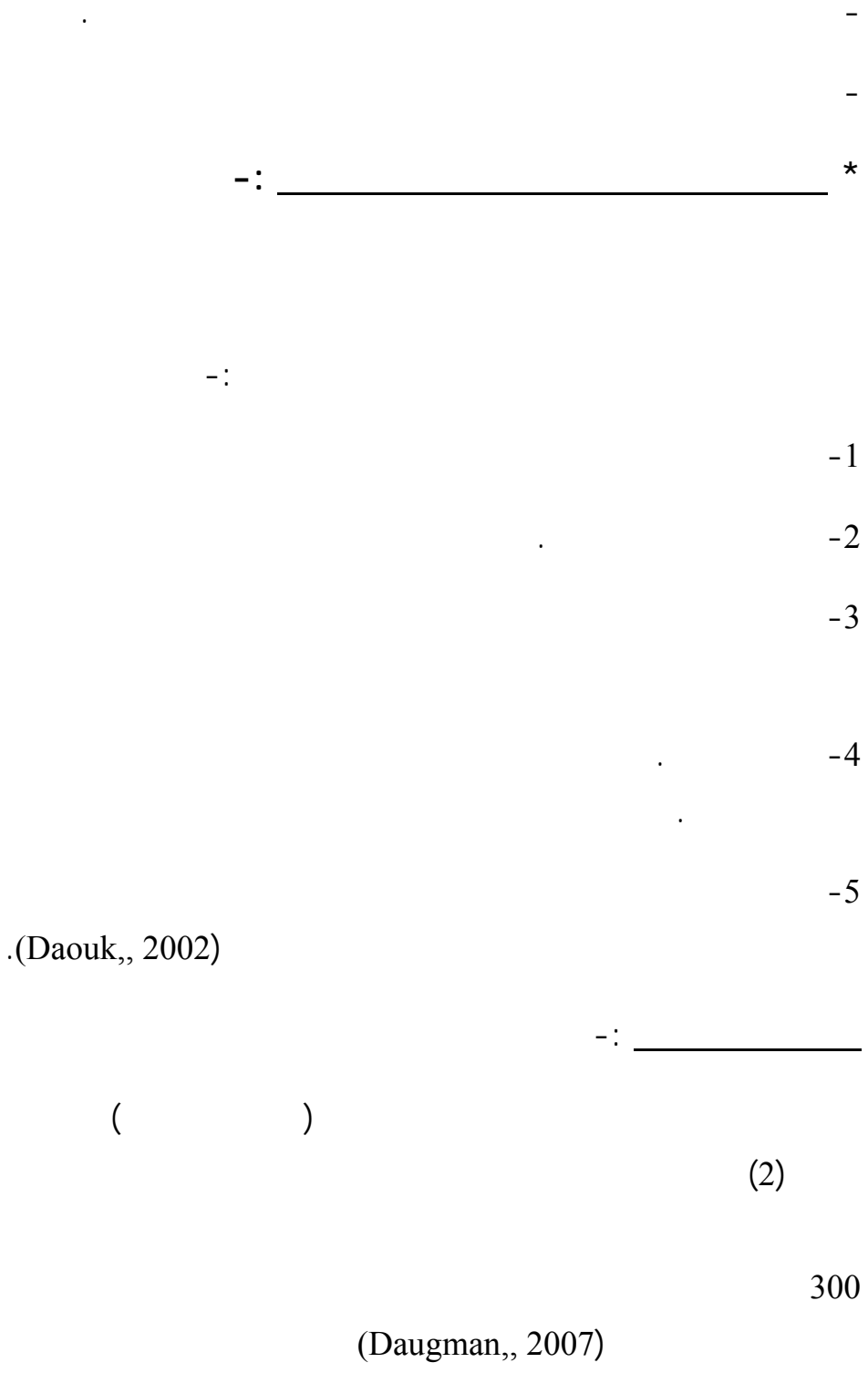
.....



(Daouk, 2002)

-()







:(2)

arrays

"templates" 512
Image processing
data base

266

512

[157]

2012 (22)

30 8

.(2007)

-: _____

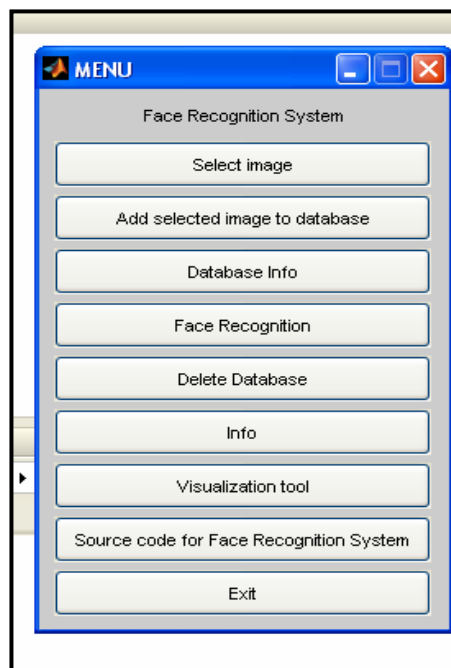
(30)

MATLAB

.(3)

MATLAB

:



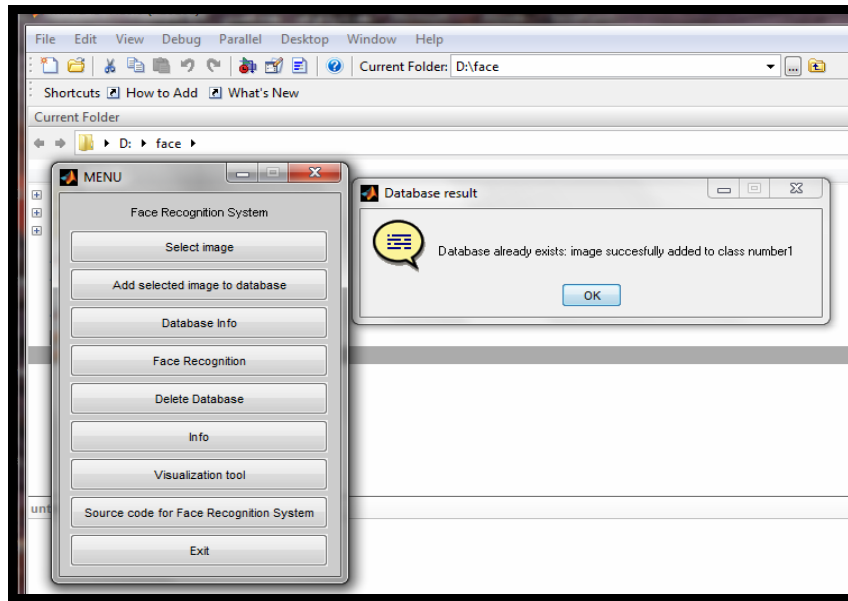
:(3)

- : Select Image _____ :

Add selected Image to _____ :
Database

(4)

(1)



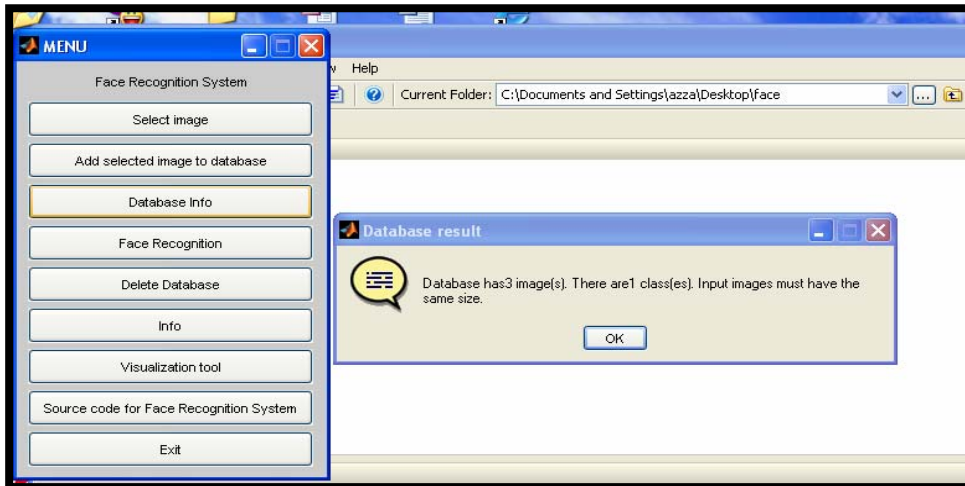
:(4)

-: **Database Info**

:

(5)

:(5)

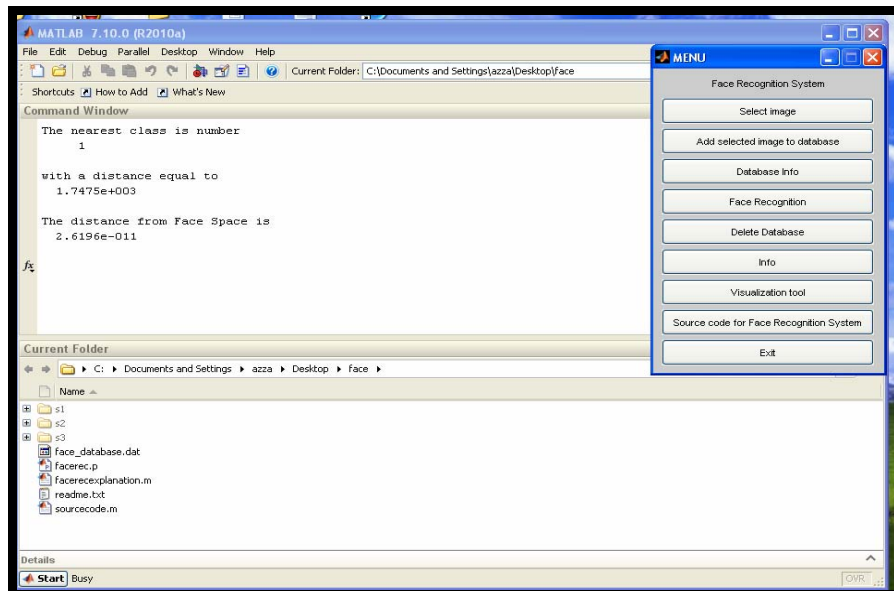


-: Face recognition :

MATLAB

.(6)

(0)



:(6)

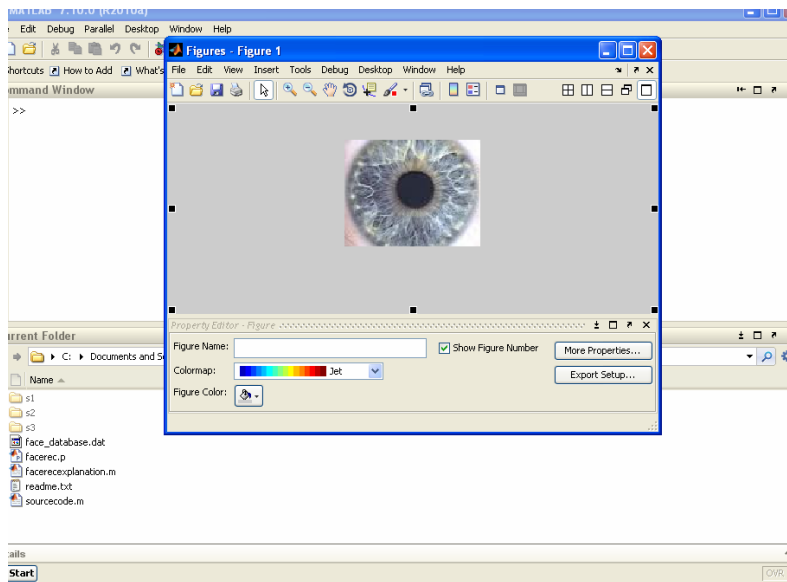
-: Delete Database :

-: Info :

. Mat lab

-: Visualization tool :

.(7)



(7): شاشة ادوات الصورة

.....
Source Code for face الاختيار الثامن: رقم الاتصال للحصول على النظام
-:Recognition

. ()

-: **Exit** _____ :

. MATLAB

-: _____

%93

30

-1

-2

-: _____

-1

-2

-: _____

.(2007) .

-1

.(2009) .

-2

- .(1982) . -3
- .(2007) . -4
- THE FREE , Wikipedia , Iris -5
recognition , encyclopaedia .
- .(2005) . -6
- 7- Daugman, J.(2007). "New methods in iris recognition."
IEEE Trans. Systems .
- 8- Masek, Libor. (2003). Recognition of Human Iris
Patterns for Biometric Identification, The University of
Western Australia.
- 9- Daouk, C. H; El-Esber, L. A.; Kammoun. F. D. and Al
Alaoui. M. A. (2002). IRIS RECOGNITION. Electrical
and Computer Engineering Department, American
University of Beirut.
- 10-Christel-loïc TISSE, Lionel MARTIN, Lionel TORRES,
Michel ROBERT. (2010) . Person identification technique
using human iris recognition, Université de Montpellier,
France

- : _____

This part gives the algorithm. The outputs are six subfigures shown in the same figure:

1. Parameters of edge detecting filters: X-axis direction filter: $Nx1=10; Sigmax1=1; Nx2=10; Sigmax2=1; Theta1=\pi/2;$ Y-axis direction filter: $Ny1=10; Sigmay1=1; Ny2=10; Sigmay2=1; Theta2=0,$

2. The thresholding parameter Alfa : $Alfa=0.1;$ Get the initial image `lena.gif` `[x, map]=gif read('lena.gif');` `w=ind2gray(x, map)` `figure(1); color map(gray)` `subplot(3,2,1), images c(w,200), title('Image: lena.gif'),` X-axis direction edge detection `subplot(3,2,2), Filter`

`x=d2dgauss(Nx1,Sigmax1,Nx2,Sigmax2,Theta1); Ix= conv2(w ,filter x, 'same ');` images (Ix),title('Ix'), Y-axis direction edge detection `subplot(3,2,3) filter` `y=d2dgauss(Ny1,Sigmay1,Ny2,Sigmay2,Theta2),`

`I y=conv2(w ,filter , 'same'), images (I y); title('I y')` Norm of the gradient (Combining the X and Y directional derivatives) `subplot(3,2,4), NVI=sqrt (Ix.*Ix +I y.*I y); images (NVI), title('Norm of Gradient')` Thres holding `I _max=max(max(NVI)), I_ min=min(min(NVI))`

`level=Alfa*(I_ max-I_ min)+I_ min; subplot(3,2,5), I b w=max(NVI ,level.*ones(size(NVI))), images c (I b w), title('After Thresholding'),`

Thinning (Using interpolation to find the pixels where the norms of

gradient are local maximum.) `subplot(3,2,6 , [n ,m]=size(I b w); for I=2: n-1, for j=2:m-1, if I b w (I , j) > level, X=[-1,0,+1;-1,0,+1;-1,0,+1];`

`Y=[-1,-1,-1;0,0,0;+1,+1,+1]; Z=[I b w (i-1,j-1),I b w(i-1,j),I b w(i-1,j+1);`

`I b w (i,j-1),I b w (I , j),I b w(i,j+1), I b w(i+1,j- 1),I b w (i+1,j) , I b w (i+1,j+1)];`

`XI=[Ix(I ,j)/NVI(I ,j) , -Ix(I ,j)/NVI(I ,j)];`

`YI=[I y (I ,j)/NVI(I ,j) , -I y(I ,j)/NVI(I ,j)];`

`ZI=interp2(X,Y,Z,XI,YI);`

`if I b w (I , j) >= ZI(1) & I b w(I ,j) >= ZI(2) I_ temp (I ,j)=I_ max ;`

`else I_ temp (I ,j)=I_ min; end else I_ temp(I ,j)=I_ min; end images c(I_ temp);` `title('After Thinning');` `color map(gray),` End of the main .m file

The functions used in the main .m file Function "d2dgauss.m":

This function returns a 2D edge detector (first order derivative

of 2D Gaussian function) with size $n1*n2$; theta is the angle that the detector rotated counter clockwise; and sigma1 and sigma2 are the standard deviation of the Gaussian functions.

```
function h = d2dgauss(n1,sigma1,n2,sigma2,theta)
```

```
r=[cos(theta) -sin(theta);
```

```
sin(theta) cos (theta)];
```

```
for I = 1 : n2
```

```
for j = 1 : n1
```

```
u = r * [j-(n1+1)/2 I -(n2+1)/2]';
```

```
h(I ,j) = gauss(u(1),sigma1)*dgauss(u(2),sigma2);
```

```
end
```

```
end
```

```
h = h / sqrt(sum(sum(abs(h).*abs(h))));
```

```
Function "gauss.m":
```

```
function y = gauss(x ,std)
```

```
y = exp(-x^2/(2*std^2)) / (std*sqrt(2*pi));
```

```
% Function "dgauss. m"(first order derivative of gauss function):
```

```
function y = dgauss(x ,std)
```

```
y = -x * gauss(x ,std) / std^2;
```

```
end of the functions
```

- clear;
- % Parameters of the Gaussian filter:
- n1=10;sigma1=3;n2=10;sigma2=3;theta1=0;
- % The amplitude of the noise:
- noise=0.1;
- [w, map]= gif read ('lena.gif');
- x=ind2gray(w ,map);
- filter1=d2dgauss(n1,sigma1,n2,sigma2,theta);
- x_rand=noise*rand n (size(x));
- y=x +x _rand;
- f1=conv2(x,filter1,'same');
- rf1=conv2(y,filter1,'same');

-
 - figure(1);
 - subplot(2,2,1);images c(x);
 - subplot(2,2,2);images c(y);
 - subplot(2,2,3);images c(f1);
 - subplot(2,2,4);images c(rf1);
 - color map(gray);
 - End of the main .m file
 - The functions used in the main. m file
 - Function "d2gauss.m":
 - This function returns a 2D Gaussian filter with size $n1*n2$; theta is
 - the angle that the filter rotated counter clockwise; and sigma1 and sigma2
 - are the standard deviation of the Gaussian functions.
 - function h = d2gauss(n1,std1,n2,std2,theta)
 - r=[cos(theta) -sin(theta);
 - sin(theta) cos(theta)];
 - for I = 1 : n2
 - for j = 1 : n1
 - u = r * [j-(n1+1)/2 i-(n2+1)/2]';
 - h(I ,j) = gauss(u(1),std1)*gauss(u(2),std2);
 - end
 - end
 - h = h / sqrt(sum(sum(h.*h)));
 - Function "gauss .m":
 - function y = gauss(x ,std)
 - y = exp(-x^2/(2*std^2)) / (std*sqrt(2*pi));
 - end of the functions