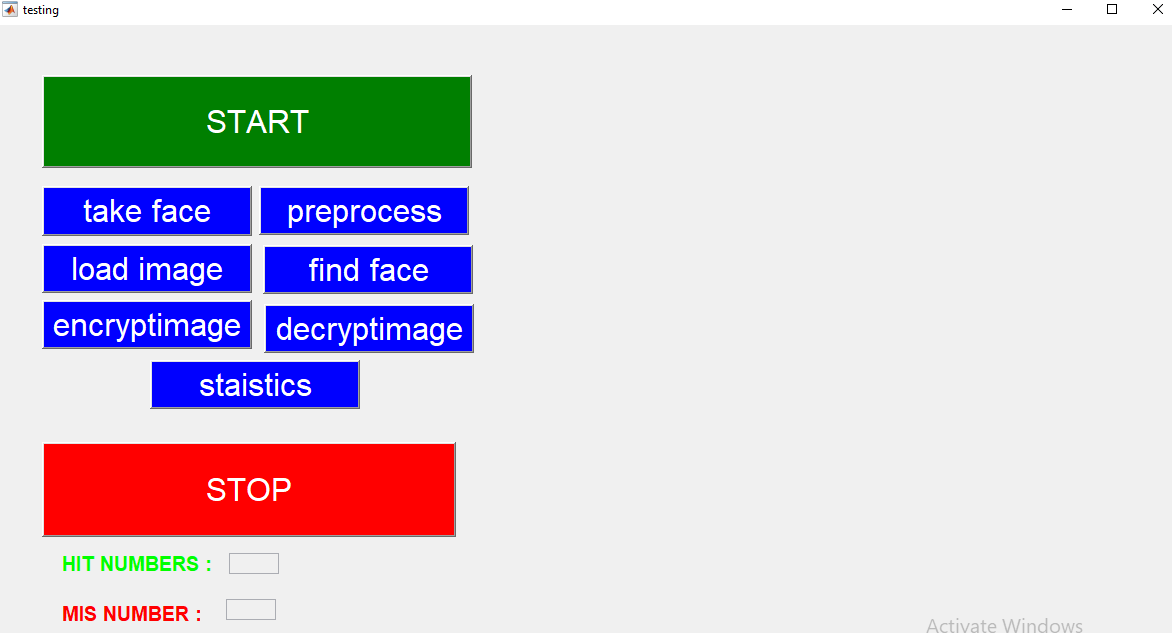
***شرح التطبيق العملي :***

***شرح مبسط لإلية عمل الكود:***



تتكون الواجهة الرئيسة للتطبيق من معرض صور هو مخفي لعدم اضافة اي صورة axes

و عدة أزرار سيتم شرح كل زر منهم على حذا

***START***

يقوم هذا الزر بتشغيل الكمرة وتهيئة الاتصال مع الواجهة الخاصة بكمرة الحاسوب المتصلة

***TAKE FACE***

يقوم هذا الزر باستخدام الاتصال الذي تم إنشائه في الزر السابق ويقوم بعرض الكمرة على مستعرض الصور AXES والكود في الأسفل هو كود هذا الزر:

function face\_Callback(hObject, eventdata, handles)

% hObject handle to face (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

%handles.vid = videoinput('winvideo' , 1, 'YUY2\_640X480');

triggerconfig(handles.vid ,'manual');

set(handles.vid, 'TriggerRepeat',inf);

set(handles.vid, 'FramesPerTrigger',1);

handles.vid.ReturnedColorspace = 'rgb';

handles.vid.Timeout = 5;

start(handles.vid);

while(1)

facedetector = vision.CascadeObjectDetector;

trigger(handles.vid);

handles.im = getdata(handles.vid, 1);

facedetector.MergeThreshold = 7 ;

bbox = step(facedetector, handles.im);

hello = insertObjectAnnotation(handles.im,'rectangle',bbox,'Face')

[m, n]=size(bbox);

if(m>0)

J = imcrop(handles.im, bbox(1, :));

save 'image.mat' J;

end

imshow(hello);

end

guidata(hObject, handles);

اولا قمنا هنا بوضع قادح ليلاحظ عند عمل الكمرا ومن ثم قمنا بالدخول بحلقة محققة دوما داخل هذه الحلقة قمنا بتطبيق خوارزمية viola jones على الصورة المأخوذة من الكمرا في الوقت الحقيقي اي مباشرة من الكمرة عن طريق اخذ فريم كل جزء من الثانية و التحقق من وجود فيدو قمنا بهذا لغرضين الاول: هو التحقق الدائم من وجود وجه والثاني هو تتبع فعند تحريك الوجه يعيد تحديد الوجه مجداً

ثم قمنا بالتحقق اي ازا لاحظ وجه فقمنا بقص هذا الوجه وحفظه في القاعدة image ليتم تطبيق خوارزمية التعرف عليه

***PREPROCESS IMAGE***

يقوم هذا الزر بمعالجة الصورة لقص الوجه منها لكي تكون دقة التعرف كبيرة لأننا قمنا بأخذ الوجوه فقط من قاعدة البيانات فعند البحث عن وجه قبل معالجة الصورة ستكون الدقة اقل ومن الممكن إن لا يتعرف النظام على هذه الصورة والكود مبين في الأسفل:

function pushbutton7\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton7 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

preprocess(handles.im)

guidata(hObject, handles);

يقوم هنا باستدعاء التابع مع تمرير الصورة المحفوظة له

كود التابع:

function preprocess(img)

imshow(img);

FaceDetect = vision.CascadeObjectDetector;

FaceDetect.MergeThreshold = 7 ;

BB = step(FaceDetect, img);

for i = 1 : size(BB,1)

rectangle('Position', BB(i,:), 'LineWidth', 3, 'LineStyle', '-', 'EdgeColor', 'r');

end

J = imcrop(img, BB(1, :));

save image.mat J;

أولا قمنا بعرض الصورة للتأكد من استقبال الصورة الصحيحة ثم قمنا بتنفيذ خوارزمية فيولا جونس على الصورة طبعا قمنا هنا بذلك لأنه يوجد إمكانية في الكود لإضافة صورة خارج الكمرة كون كمرة الحاسوب تكون غير واضحة وغير صافية قد ينتج بعض الأخطاء لذلك أضفت زر لتحميل صورة من الحاسوب فهذه الصورة يجب اعادة معالجتها قبل البحث عن الوجه وكما نلاحظ انه في حال معالجة صورة لاتحوي وجه سيعيد البرنامج خطأ اي الصورة لاتحوي وجه أو لم يستطع التعرف على الوجه في هذه الحالة نقوم بالبحث عن الصورة دون إعادة معالجتها ففي حال كانت تحوي وجه سيعطي نتيجة أو سيعطي انه غير معرف

***LOAD IMAGE***

هذا الزير يحمل صورة من الحاسوب ويعرضها علة المستعرض ويقوم بتخزين الصورة في القاعدة image احتياطاً في حال قام المستخدم في البحث عن وجه دون عملية المعالجة المسبقة

function pushbutton8\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton8 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

[f,p]=uigetfile('\*.\*','select image to test');

ff=strcat(p,f);

img=imread(ff);

imshow(img);

handles.im=img;

J=img;

save 'image.mat' J;

handles.output = hObject;

guidata(hObject, handles);

***FIND FACE***

يقوم هذا الزر في استخراج الميزات من الصورة المراد البحث عنها في قاعدة المعطيات ومن ثم يقارن هذه الميزات مع مصفوفة الميزات الموجودة لدينا وفي حال كان اقل فارق بين ميزات الصورة المراد البحث عنها وصور القاعدة أدنى من معدل خطأ معين نقبل الصورة ونعرض الصورة مع الصورة التي تعرف عليها أما إذا كانت اكبر من قيمة الخطأ نعتبر الشخص غير معروف

function pushbutton9\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton9 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

load image.mat;

test(J);

كود التابع test

function test(img)

load pcadb;

imgo=img;

img=rgb2gray(img);

img=imresize(img,[M,N]);

img=double(reshape(img,[1,M\*N]));

imgpca=(img-m)\*Ppca;

distarray=zeros(n,1);

for i=1:n

distarray(i)=sum(abs(T(i,:)-imgpca));

end

[r,index]=min(distarray);

if r>1.8e+04

msgbox('not found');

else

figure(2);

rimg=imread(sprintf('./traindb/%d.jpg',index));

subplot(121)

imshow(imgo);

title('query face');

subplot(122)

imshow(rimg);

title('recognized face');

end

***encryptimage***

يقوم هذا الزر بتشفير الصورة باستخدام خوارزمية rsa فيقوم باستدعاء التابع rsa ويمرر له الصورة المراد تشفيرها ويظهر رسالة تبين انتهاء عملية التشفير

% --- Executes on button press in pushbutton10.

function pushbutton10\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton10 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

load image.mat;

msgbox('this will take some time...')

rsa(J);

msgbox('image encrypted successfully');

guidata(hObject, handles);

أما بالنسبة لكود التابع rsa فهو كالاتي :

function rsa(img)

import javax.crypto.Cipher;

plaintext = img;

plaintext= imresize(plaintext,[400,360]);

imshow(plaintext)

[M,N,Z]=size(plaintext)

plaintext1=reshape(plaintext(:,:,1),[1,M\*N]);

plaintext2=reshape(plaintext(:,:,2),[1,M\*N]);

plaintext3=reshape(plaintext(:,:,3),[1,M\*N]);

p1=[]

p2=[]

p3=[]

for i=1:2880

b=i-1;

p1=[p1 ;[ plaintext1((b\*50)+1:(50\*i))]];

p2=[p2 ;[ plaintext2((b\*50)+1:(50\*i))]];

p3=[p3 ;[ plaintext3((b\*50)+1:(50\*i))]];

end

cipher = Cipher.getInstance('RSA');

keygen = java.security.KeyPairGenerator.getInstance('RSA');

keyPair = keygen.genKeyPair();

cipher.init(Cipher.ENCRYPT\_MODE, keyPair.getPrivate());

keyPair.getPrivate()

keyPair.getPublic()

plaintextUnicodeVals1=[];

plaintextUnicodeVals2=[];

plaintextUnicodeVals3=[];

plaintextBytes2=[];

plaintextBytes3=[];

plaintextBytes1=[];

ciphertext1=[];

ciphertext2=[];

ciphertext3=[];

for i=1:2880

plaintextUnicodeVals1=[plaintextUnicodeVals1 ;int8(p1(i,:))];

plaintextUnicodeVals2=[plaintextUnicodeVals2 ;int8(p2(i,:))];

plaintextUnicodeVals3=[plaintextUnicodeVals3 ;int8(p3(i,:))];

plaintextBytes1 = [plaintextBytes1 ; typecast(plaintextUnicodeVals1(i,:), 'int8')];

plaintextBytes2 = [plaintextBytes2 ; typecast(plaintextUnicodeVals2(i,:), 'int8')];

plaintextBytes3 = [plaintextBytes3 ; typecast(plaintextUnicodeVals3(i,:), 'int8')];

ciphertext1 =[ ciphertext1 cipher.doFinal(plaintextBytes1(i,:))];

ciphertext2 =[ ciphertext2 cipher.doFinal(plaintextBytes2(i,:))];

ciphertext3 =[ ciphertext3 cipher.doFinal(plaintextBytes3(i,:))];

end

public\_key=keyPair.getPublic();

save cihperimage.mat ciphertext1 ciphertext2 ciphertext3

save key.mat public\_key

اذ نقسم الصورة الى ثلاث مصفوفات واحدة للاحمر وواحدة للاخضر وواحدة للازرق ومن ثم نقسم كل مصفوفة لمجموعة مصفوفات طول كل واحدة منها 50 عنصر لان خوارزمية rsa لاتتعامل مع بيانات اكبر من 117byte لذلك قمنا بتقسيم كل مصفوفة لمجموعة مصفوفات بعد الواحدة منها 50 واصبح لدينا ثلاث مصفوفات تحوي المعلومات المشفرة كل مصفوفة مكونة من عدة مصفوفات و المفتاح العام للتشفير

***Decryptimage***

يقوم هذا الزر بفك تشفير الصورة باستخدام المفتاح العام والمصفوفات المشفرة فقط اي دون معرفة الصورة الاصلية ويعيد تشكيل الصورة الاصلية

% --- Executes on button press in pushbutton11.

function pushbutton11\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton11 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

dec\_rsa;

guidata(hObject, handles);

اما بالنسبة لتابع dec\_rsa

function dec\_rsa

import javax.crypto.Cipher;

load cihperimage.mat;

load key.mat;

cipher = Cipher.getInstance('RSA');

cipher.init(Cipher.DECRYPT\_MODE, public\_key);

decryptedBytes1=[];

decryptedBytes2=[];

decryptedBytes3=[];

decryptedText1=[];

decryptedText2=[];

decryptedText3=[];

for i=1:2880

decryptedBytes1 =[decryptedBytes1 cipher.doFinal(ciphertext1(:,i)) ];

decryptedText1 = [decryptedText1 typecast(decryptedBytes1(:,i), 'int8')];

decryptedBytes2 =[decryptedBytes2 cipher.doFinal(ciphertext2(:,i)) ];

decryptedText2 = [decryptedText2 typecast(decryptedBytes2(:,i), 'int8')];

decryptedBytes3 =[decryptedBytes3 cipher.doFinal(ciphertext3(:,i)) ];

decryptedText3 = [decryptedText3 typecast(decryptedBytes3(:,i), 'int8')];

end

image1=[];

image2=[];

image3=[];

for i=1:2880

image1 =[image1 ; decryptedText1(:,i) ];

image2 =[image2 ; decryptedText2(:,i) ];

image3 =[image3 ; decryptedText3(:,i) ];

end

image1=reshape(image1,[400,360]);

image2=reshape(image2,[400,360]);

image3=reshape(image3,[400,360]);

final\_image=[];

final\_crypted\_image=[];

final\_crypted\_image(:,:,1)=ciphertext1;

final\_crypted\_image(:,:,2)=ciphertext2;

final\_crypted\_image(:,:,3)=ciphertext3;

final\_image(:,:,1)=image1;

final\_image(:,:,2)=image2;

final\_image(:,:,3)=image3;

final\_image=uint8(final\_image);

final\_crypted\_image=uint8(final\_crypted\_image);

figure(2);

subplot(121)

imshow(final\_image);

title('image');

subplot(122)

imshow(final\_crypted\_image);

title('rsa img');

هنا فقط قمنا بفك تشفير المصفوفات المشفرة باستخدام المفتاح العام والحصول على المصفوفات الثلاثة ثم إعادة جمعها لتشكيل ثلاث مصفوفات غير مقسمة ثم قمنا بوضع الثلاث مصفوفات لتشكيل الصورة الملونة rgb

***Statistics***

يقوم هذا الزر بتجريب نظام التعرف على الوجوه ل600 صورة يوجد بها بعض الصور التي قمنا بتشويهها يدوياً لاختبار دقة التعرف

% --- Executes on button press in pushbutton12.

function pushbutton12\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton12 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

msgbox('this will take some time...')

[t,f]=hitandmise();

msgbox('done!!!')

edit1 = findobj(0, 'tag', 'edit1');

edit2 = findobj(0, 'tag', 'edit2');

set(edit1, 'String', num2str (t));

set(edit2, 'String', num2str (f));

guidata(hObject, handles);

اما بالنسبة للتابع فهو كالأتي

function [true, false]=hitandmise

true=0;

false=0;

for count=1:600

I=imread(sprintf('testdb/%d.jpg',count));

preprocess(I);

load image.mat

[a b]=stat(J,count);

true=true+a;

false=false+b;

end

اذ يقوم بارسال الصورة ورقمها للتابع الذي يعيد فيما اذا كان التعرف صحيح ام خاطئ ونقوم بجمع هذه القيم فيظهر لدينا بشكل نهائي عدد مرات التعرف الصحيح وعدد مرات التعرف الخاطئ

تابع stat

function [true false] =stat(img,number)

load pcadb;

imgo=img;

img=rgb2gray(img);

img=imresize(img,[M,N]);

img=double(reshape(img,[1,M\*N]));

imgpca=(img-m)\*Ppca;

true=0;

false=0;

distarray=zeros(n,1);

for i=1:n

distarray(i)=sum(abs(T(i,:)-imgpca));

end

[r,index]=min(distarray);

if r>1.8e+04

false=false+1;

else

true\_n=ceil(number/20);

b=ceil(index/10);

if(true\_n==b)

true=true+1;

else

false=false+1;

end

end

r

هو مشابه لتابع test لكن هنا نقوم باختبار رقم الصورة بعد تقسيمها على عدد الصور للشخص الواحد فينتج لدينا رقم الشخص في حال تطابق مع رقم الشخص في القاعدة نعتبر انه تم التعرف اما في حال لم يتم التعرف او تم التعرف لكن الرقم لا يتطابق اي الشخص خاطئ نعتبر انه لم يتم التعرف

***STOP***

يقوم هذا الزر بايقاف الكمرة كما يقوم بعرض الصورة بعد معالجتها في المستعرض

function stop\_Callback(hObject, eventdata, handles)

% hObject handle to stop (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

load image.mat;

imshow(J);

handles.output = hObject;

stop(handles.vid),clear handles.vid

guidata(hObject, handles);

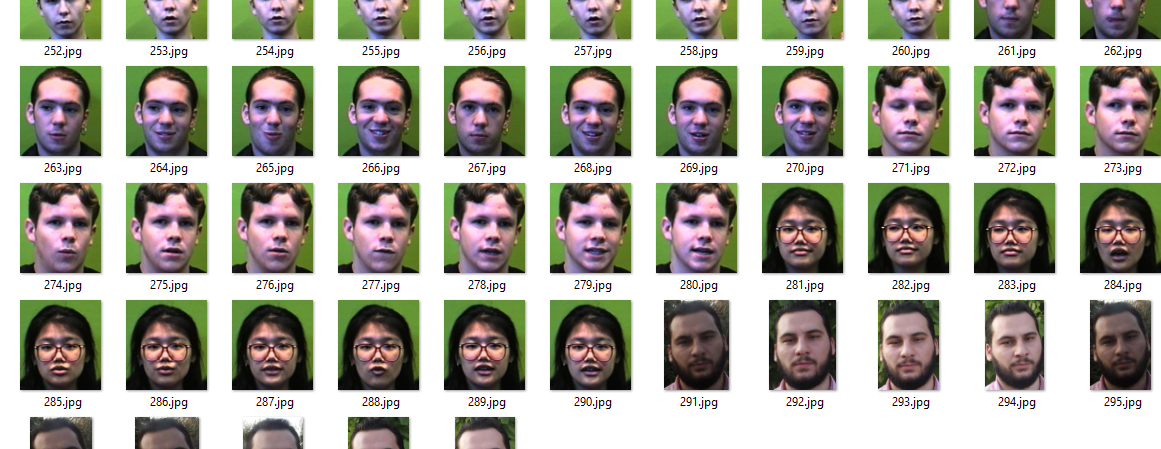
اما بالنسبة لقاعدة البيانات فقد استعنت بالقاعدة faces94 وهي قاعدة تحوي 20 صورة لكل شخص وعدة اشخاص ذكور واناث لكن المشكلة التي واجهناها هي ان هذه الصور ملونة والتعامل يجب ان يكون مع الصور الرامدية لذلك نقوم بتحويلها لصور رمادية وقمنا بتقليل ابعاد الصورة وذلك لان الحجم عند اخذ 300 صورة يكون ضخم فثبتنا الابعاد على 100\*90

وبالنسبة للميزات فقد استخدمنا خوارزمية pca في تقليص عدد الميزات واخذنا افضل 50 مميز لكل صورة وخزناها في مصفوفة المميزات ومن ثم خزنا جميع المتحولات في قاعدة اسمها pcadb لان العملية تستغرق وقت اي عملية المعالجة لقاعدة البيانات فعند البحث عن صورة نقارنها مع هذه البيانات

رابط القاعدة :

<https://cswww.essex.ac.uk/mv/allfaces/faces94.html>

اما بالنسبة للصور فقد اعدنا تسميتها من1 الى 300 ليسهل التعامل معها ضمن الماتلاب ضمن ملف traindb



اما كود التدريب فهو كالاتي

M=100;

N=90;

n=300;

L=50;

X=zeros(n,(M\*N));

T=zeros(n,L);

for count=1:n

I=imread(sprintf('%d.jpg',count));

preprocess(I);

load image.mat

I=rgb2gray(J);

I=imresize(I,[M,N]);

X(count,:)=reshape(I,[1,M\*N]);

end

xb=X;

m=mean(X);

for i=1:n

X(i,:)=X(i,:)-m;

end

q=(X'\*X)/(n-1);

[Evecm,Evalm]=eig(q);

Eval=diag(Evalm);

[Evalsorted,index]=sort(Eval,'descend');

Evecsorted=Evecm(:,index);

Ppca=Evecsorted(:,1:L);

for i=1:n

T(i,:)=(xb(i,:)-m)\*Ppca;

end

ونلاحظ اننا قد قمنا باعداة تشكيل مصفوفة الصورة من ثنائة البعد M,N الى احادية البعد بعدها 1,M\*N وذلك لان خوارزمية pca تتعامل مع مصفوفات احادية البعد

كما قمنا ب طرح متوسط الصور من الصور وذلك لتحسين الدقة فهذه القيم موجودة في كل الصور وهي عبارة عن شكل عيون وانف وفم فهي موجودة في كل الصور لذلك لادعي لها بما اننا نريد معرفة الفروقات بين الصور فقط

لدينا التابع

***compare\_two\_image()***

الذي نقوم بتمرير صورتين له ويقوم برسم مقارنة لتغير البكسلات بين الصورة الاولى والثانية مع بعضها البعض في ثلاث منحنيات بيانية لكل لون منحني

function compare\_two\_image()

[f,p]=uigetfile('\*.\*','select image to compare data');

ff=strcat(p,f);

img=imread(ff);

[f,p]=uigetfile('\*.\*','select second image to compare data');

ff=strcat(p,f);

img2=imread(ff);

img=imresize(img,[100,90]);

img2=imresize(img2,[100,90]);

x=[];

y=[];

for i=1:100

for j=1:90

x=[x img(i,j,1)];

y=[y img2(i,j,1)];

end

end

figure(2);

plot(x,y,'--rs',...

'LineWidth',2,...

'MarkerSize',1,...

'MarkerEdgeColor','r',...

'MarkerFaceColor','r'),title('red color');

x=[];

y=[];

for i=1:100

for j=1:90

x=[x img(i,j,2)];

y=[y abs(img2(i,j,2)-img(i,j,2))];

end

end

figure(3);

plot(x,y,'--gs',...

'LineWidth',2,...

'MarkerSize',1,...

'MarkerEdgeColor','g',...

'MarkerFaceColor','g'),title('green color');

x=[];

y=[];

for i=1:100

for j=1:90

x=[x img(i,j,3)];

y=[y img2(i,j,3)];

end

end

figure(4);

plot(x,y,'--bs',...

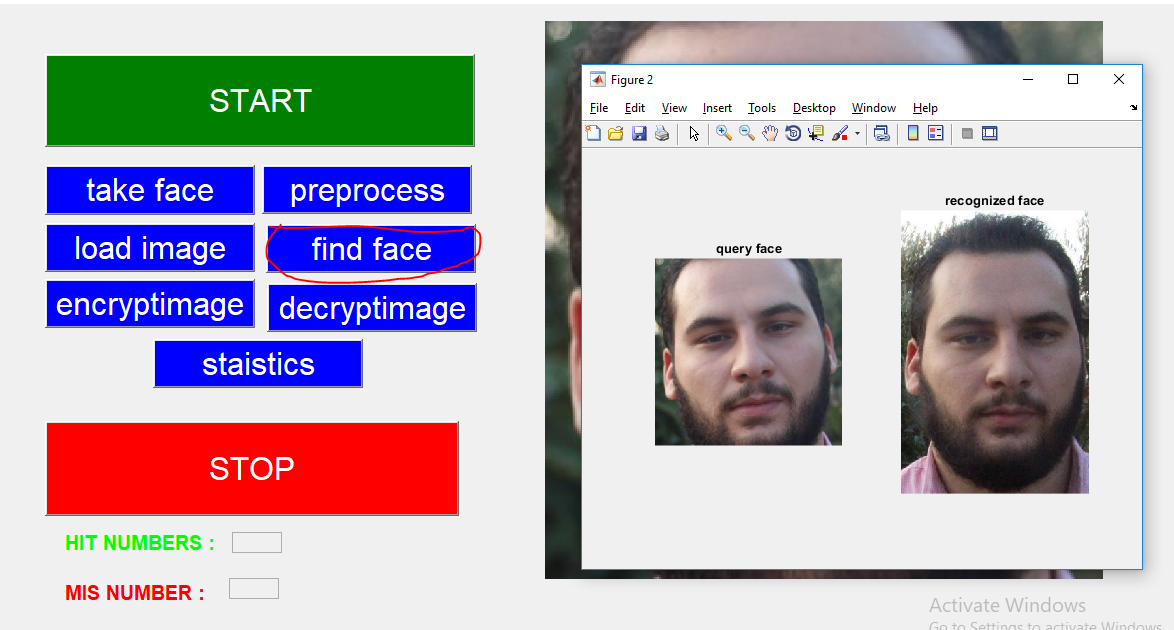
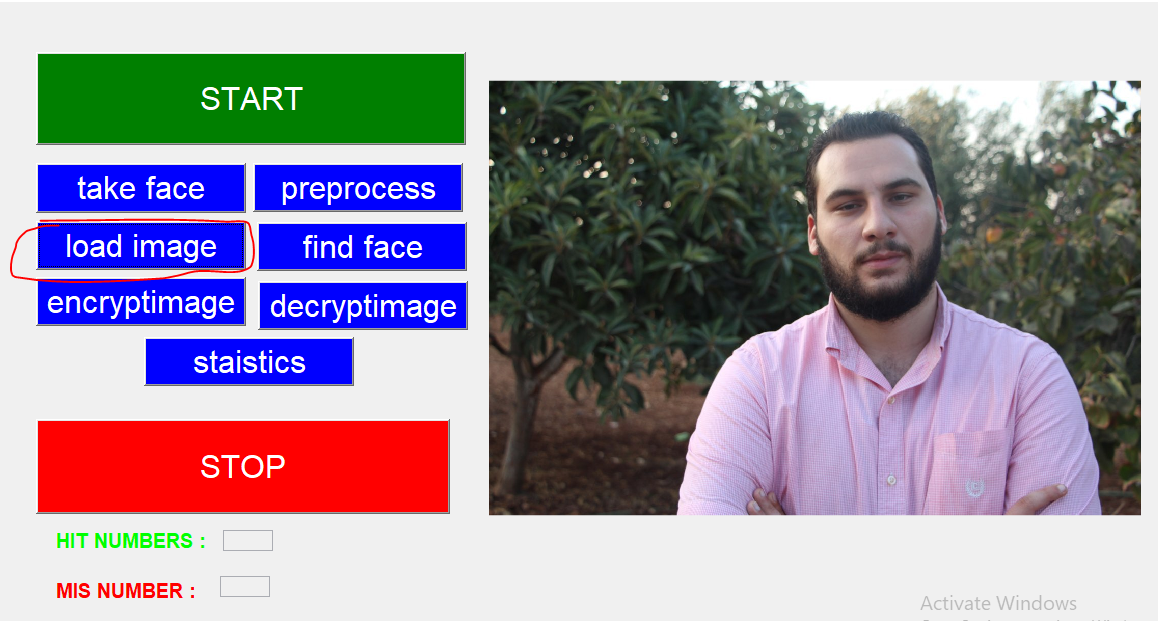
'LineWidth',2,...

'MarkerSize',1,...

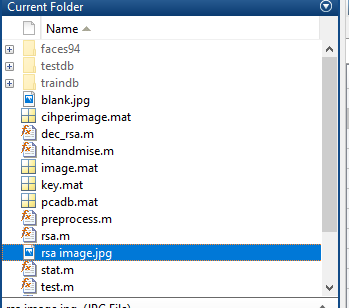
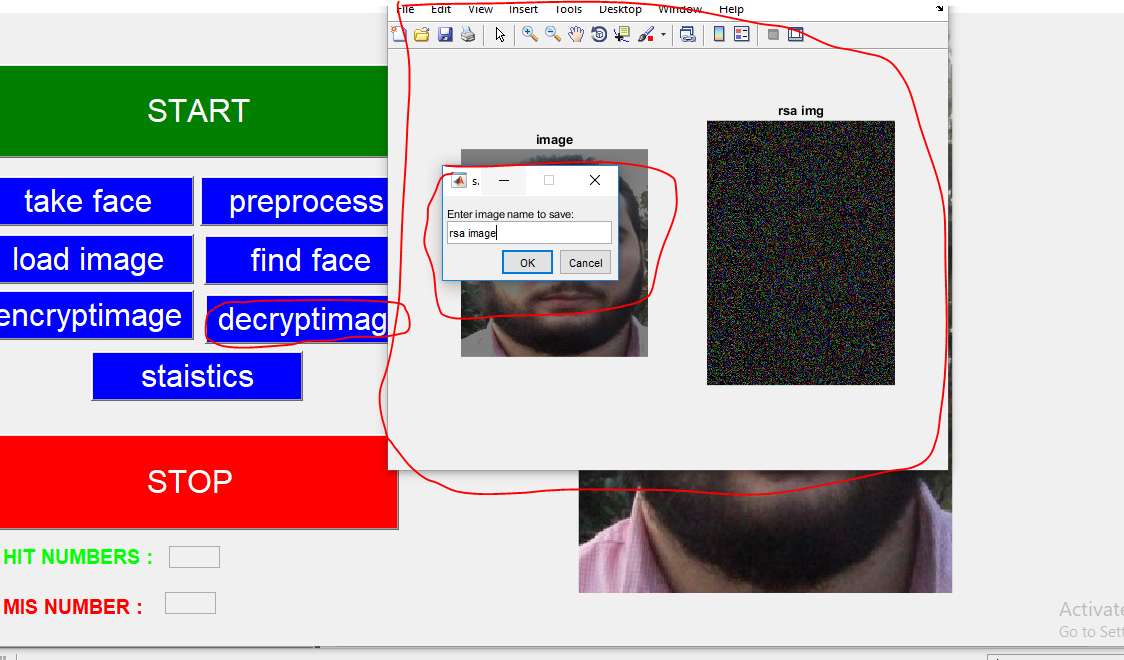
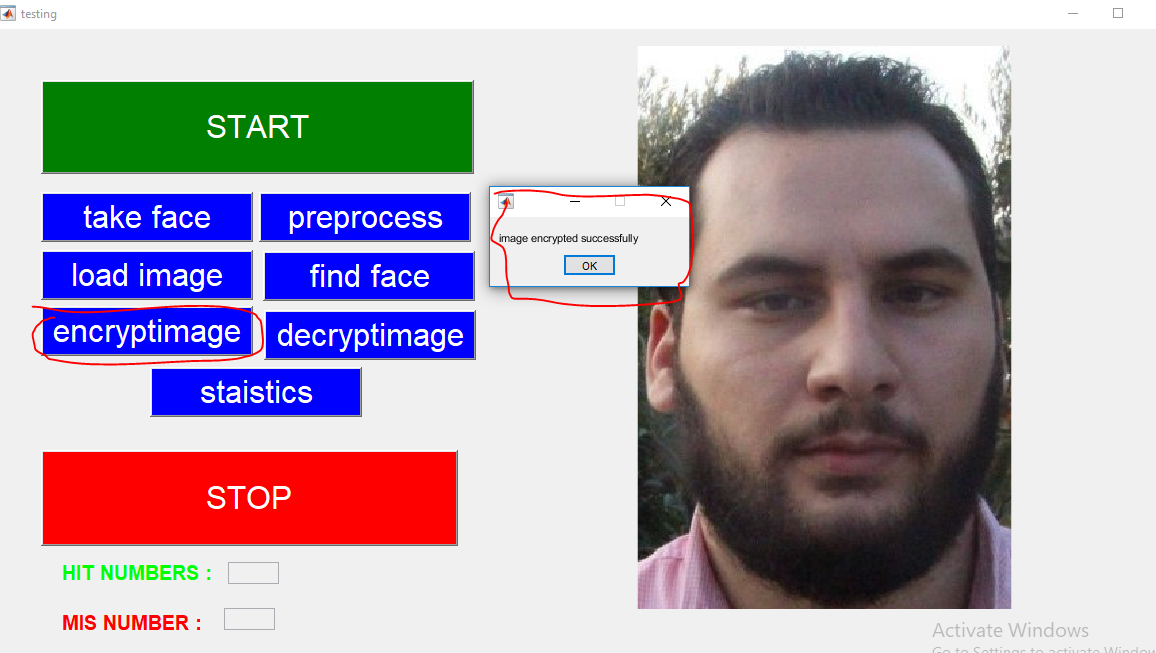
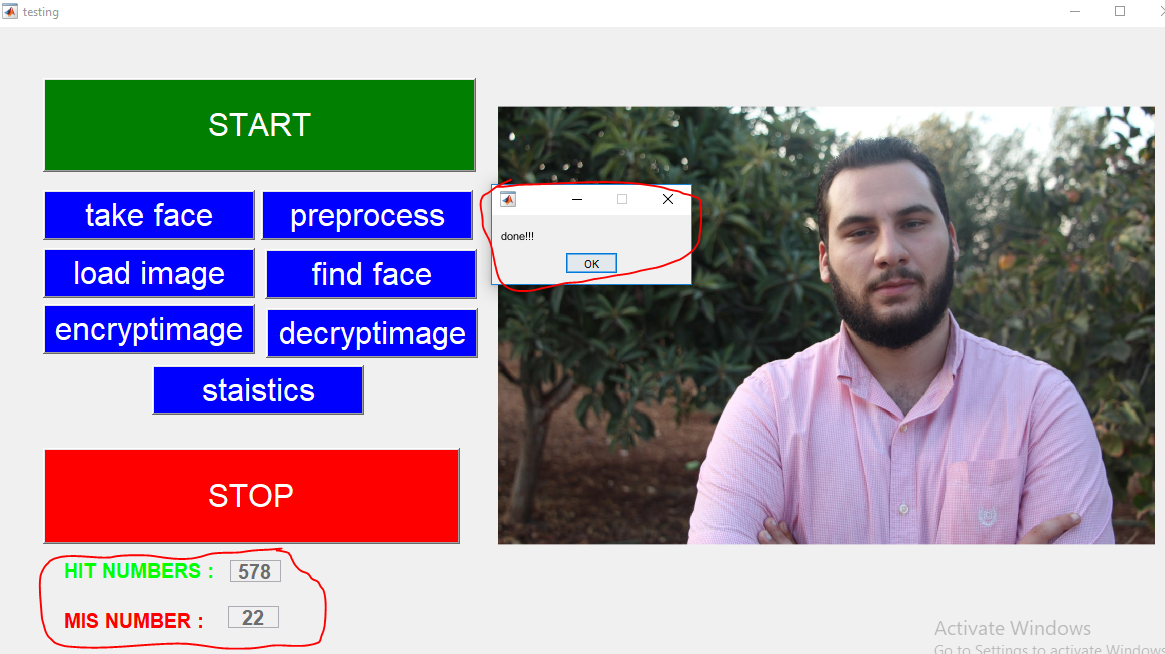
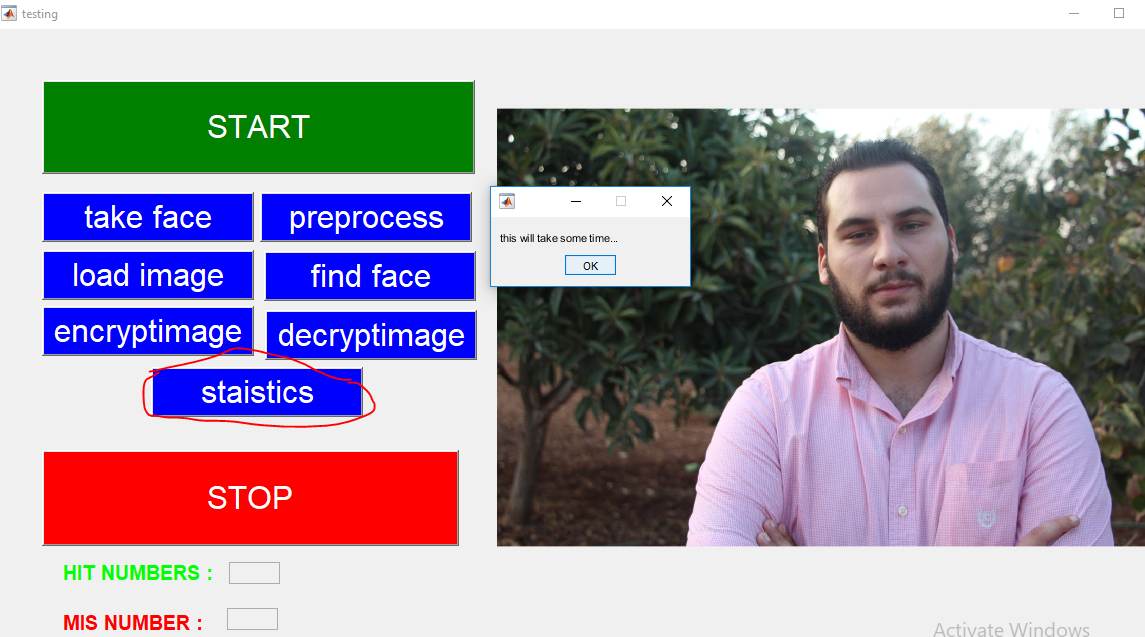
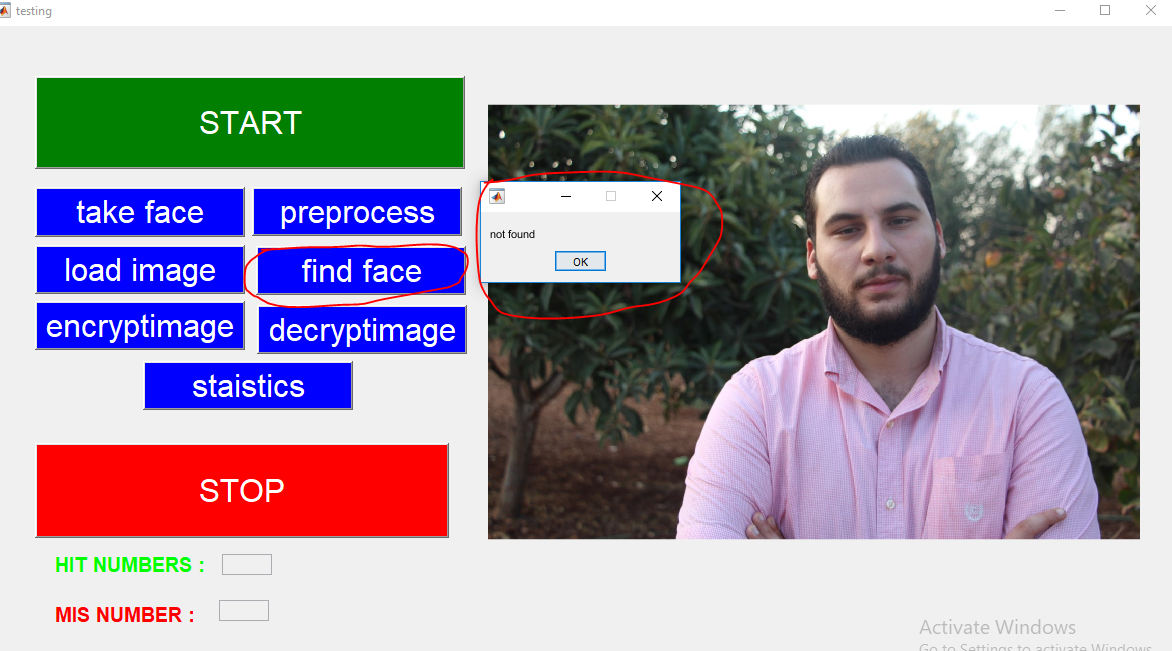
'MarkerEdgeColor','b',...

'MarkerFaceColor','b'),title('blue color');

والتنفيذ كما يلي :



اما في حال اردنا البحث دون معالجة الصورة:



عند مقارنة الصورة نفسها مع تشفيرها يكون الرسم كما يلي :

